uc3m | Universidad Carlos III de Madrid

Grado en Ingeniería Electrónica Industrial y Automática

2018-2019

Trabajo Fin de Grado

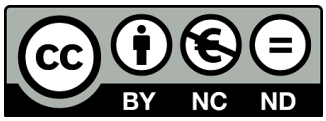
Sistema de Iluminación Inteligente para Transporte Recreativo (II)

Fátima Saghouani Ben khalek

Tutor/es

Cristina de Dios Fernández

Lugar y fecha de presentación prevista
Leganés, 5 de julio de 2019



RESUMEN

En octubre de 2018 se aprobó la Ordenanza de Movilidad Sostenible para la ciudad de Madrid, en la cual se limita la velocidad a la que pueden circular los patines y patinetes por las aceras y zonas peatonales[1].

El presente documento plantea una posible solución para los patinadores ante esta nueva normativa. Esta solución se basa en un sistema de iluminación y navegación formado por un circuito y una aplicación móvil para Android. El circuito que hará la función de sistema de iluminación se comunicará vía Bluetooth con una aplicación móvil para Android que hará la función de sistema de navegación.

El objetivo del proyecto, ademas de indicarle al usuario si puede circular por zonas peatonales o no, es hacer que circule de una manera más segura aumentado su visibilidad para otros usuarios y para si mismo y proporcionando al usuario una manera rápida para saber si esta excediendo el limite. De esta manera el usuario puede circular tranquilamente sin la duda de si estará respetando la normativa.

Desde la aplicación el usuario podrá conectarse vía Bluetooth y encender o apagar el sistema de iluminación, ademas podrá ver su velocidad actual y la ruta recorrida en un mapa. A su vez el sistema de iluminación consta de un LED rojo y uno verde que, de manera autónoma, le indicaran al usuario si la velocidad a la que circula es la adecuada para ir por zonas peatonales.

Palabras clave: LED, Raspberry Pi, Bluetooth, aplicación móvil, Android.

[1] BO. Ayuntamiento de Madrid 23/10/2018 núm. 8263 pág. 11 - 182

ABSTRACT

In October 2018, a new mobility ordinance for the city of Madrid was approved, limiting the speed at which skates and scooters can circulate along sidewalks and pedestrian zones[2].

This document proposes a possible solution for the skaters for this new regulation. This solution is based on a lighting and navigation system consisting of a circuit and a mobile application for Android. The circuit that will make the function of lighting system will communicate via Bluetooth with a mobile application for Android that will act as the navigation system.

The objective of the project, in addition to indicating to the user if it can circulate in pedestrian zones or not, is to have them circulate in a safer way, increasing their visibility for other users and for themselfs and providing the user with a quick way to know if they're exceeding the limit. This way the user can circulate quietly without the doubt of whether they will be respecting the regulations.

From the application the user can connect via Bluetooth and turn on or off the lighting system, in addition the users can see their current speed and the route traveled on a map. The lighting system consists of red and green LEDs that, in an autonomous way, will indicate to the user if the speed at which it circulates is adequate to go through pedestrian zones.

Keywords: LED, Raspberry Pi, Bluetooth, mobile application, Android.

[2]BO. Ayuntamiento de Madrid 23/10/2018 núm. 8263 pág. 11 - 182

A mis amigos, a mi familia, a mis hermanos, a mis profesores, nada de esto habría sido posible sin vosotros, sin vuestro apoyo y vuestros ánimos cuando las cosas iban cuesta arriba, ¡Gracias!

Pero sobretodo gracias a ti mamá, por tu esfuerzo constante y toda tu vida de dedicación a mi y a mis hermanos, tú siempre has sido una inspiración para mí y sabes que nada de esto habría sido posible sin ti.

Gracias mamá.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1. PRESENTACIÓN

En esta sección se comentarán los motivos y objetivos que se tienen con el desarrollo del proyecto y se hará un breve descripción del sistema que se planteado ante la problemática expuesta.

1.1. Motivación y objetivos

Desde el origen de la maquina de vapor hasta los coches eléctricos actuales los medios de transporte han sufrido grandes transformaciones ligadas a la evolución tecnológica y las necesidades del momento.

Actualmente las necesidades y preocupaciones respecto al transporte son varias, en 2015 4000 millones de personas vivía en ciudades y se estima que en 2030 este numero aumentara hasta llegar a 5000 millones[1], este crecimiento en la población urbana afecta directamente al transporte. Poniendo como ejemplo la ciudad de Madrid, día a día se puede observar como en horas puntas, el centro de la cuidad se encuentra paralizado, trenes, metro y autobuses en sus máximos de ocupación y las carreteras colapsadas debido al volumen de automóviles.

Otro de los problemas del transporte es la contaminación. Aunque las ciudades ocupan tan solo el 2 % de la superficie terrestre son las causantes de mas del 60 % de las emisiones de CO_2. La OMS establece ciertas normas de calidad del aire las por el bienestar de los ciudadanos, desde 2016 el 90 % de la población respiraba aire que no cumplía dichas normas de seguridad[1].

El siguiente paso de los medios de transporte es la reducción de la emisiones y la automatización de la conducción, pero aunque se creen automóviles eléctricos eficientes y autónomos esto no solventa la problemática surgida del aumento de la población en las ciudades. La tendencia actual para enfrentar esta situación es cambiando el modo en el los cuidadanos se mueven por las ciudades, cada vez se ve a mas gente por Madrid circulando en bicicletas y patinetes y se han creado numerosas empresas de alquiler de estos medios.

El incremento en el uso de medios de transporte como bicicletas y patinetes ha provocado que le ayuntamiento de Madrid intervenga en la forma de circular de estos, no solo para su propia seguridad sino también por la comodidad y seguridad de los peatones y demás usuarios de la vía publica. Desde el 24 de octubre de 2018 está en vigor en la cuidad de Madrid la Ordenanza de Movilidad Sostenible. Esta normativa establece diferentes reglas para los patinetes y bicicletas mecánicos y para los motorizados, los llamados Vehículos de Movilidad Urbana (VMU).

Esta normativa surge del aumento de los niveles de contaminación en el centro de Madrid, por ello a la hora de elaborar la normativa ha primado fomentar los transpor-

tes sostenibles como las bicicletas y patines reforzando las normas para conseguir una convivencia pacifica con los peatones.

Para los patines y patinetes sin motor, ahora es obligatorio circular a menos de diez kilómetros por hora por aceras y zonas peatonales. Para circular por carriles bici no protegidos los patinadores deberán circular con casco y señalizados con elementos reflectantes, pero la novedad mas difícil de adoptar es que para casos de visibilidad reducida deberán circular con luces de posición, he aquí donde entra este proyecto.

Los transportes alternativos para cuidad son un nuevo mercado en auge. El aumento de población y la contaminación no hacen mas que fomentar el uso de estos vehículos y a pesar de que las normativas intentan ponerse al día la burocracia lo hace complicado. Por eso nosotros planteamos un sistema de iluminación y seguimiento que aumentará la seguridad de los usuarios a la par que ayudarles a motorizar su actividad, haciendo un seguimiento de la ruta que recorren y de la velocidad que alcanzan mediante un circuito electrónico y una aplicación móvil.

1.2. Descripción de la solución

El presente documento plantea parte de la solución propuesta para ciclistas y patinadores, no solo ante la nueva normativa para la ciudad de Madrid, sino para circular de manera mas segura y cómoda.

La solución diseñada ha sido dividida en dos proyectos que se han llevado a cabo en paralelo. El primero, llamado Sistema de Iluminación Inteligente para Transporte Recreativo (I), en adelante SIITR (I), se ha encargado del diseño e implementación de toda la electrónica necesaria para el sistema de Iluminación. La segunda parte, a la que refiere el presente documento, es la encargada del diseño del Sistema de Navegación y la conexión entre ambos sistemas.

El proyecto, como un global, obtiene información de la velocidad del patinador y de manera autónoma ilumina un piloto rojo o verde. El pilo rojo le indica al patinador que la velocidad a la que circula no es la adecuada para ir junto a peatones, en este caso el patinador deberá reducir su velocidad hasta que el piloto rojo se apague y se encienda el verde, que indica que circula a una velocidad segura. Además el sistema de iluminación dispondrá de luces de posición y luces para mejorar la visibilidad del patinador.

La velocidad actual del patinador se envía a una aplicación Android -que hace la función de Sistema de Navegación- que está conectada vía Bluetooth al Sistema de Iluminación. La aplicación muestra por pantalla el dato de velocidad del usuario a la vez que muestra un mapa en el que se traza, en tiempo real, la ruta recorrida por el patinador. Además mediante la aplicación se puede controlar la conexión y el encendido y apagado el Sistema de Iluminación para que, en caso de no estar circulando, se puedan apagar luces.

Al nivel esquemático el funcionamiento el proyecto es el siguiente:

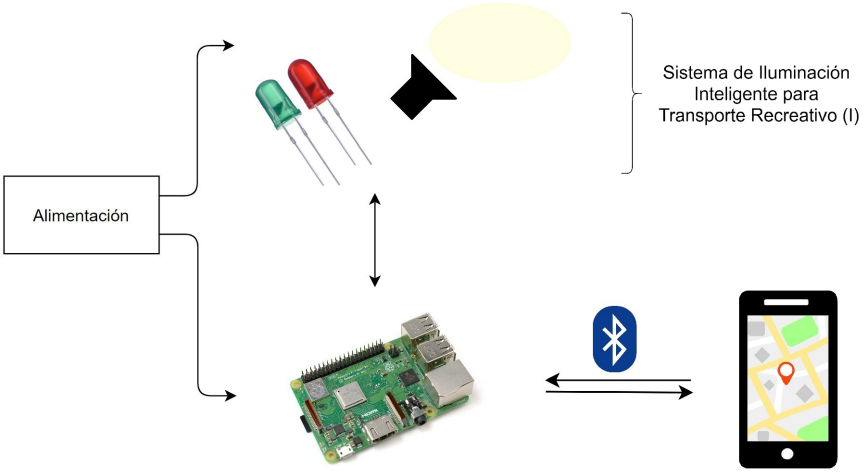


Fig. 1.1. Diagrama de funcionamiento del sistema

El Sistema de Iluminación se conecta con una Raspberry Pi que se encarga de procesar la información y compartirla vía Bluetooth con el Sistema de Navegación. Este Sistema de Navegación es el formado por el dispositivo móvil al cual se le instala la aplicación Android comentada anteriormente.

2. GESTIÓN DEL PROYECTO

En esta sección se especificarán los detalles administrativos del proyecto, como la planificación, presupuesto, normativas y demás aspectos que afectan al desarrollo del proyecto.

2.1. Planificación

En este apartado se presenta una tabla de planificación (ver Tabla 2.1) del proyecto con la diferentes metas que se han logrado, una pequeña descripción del objetivo que se debe alcanzar en cada meta para considerarla como cumplida, la tecnología que se ha de trabajar para dicha meta, y una estimación en jornadas laborales.

La estimación se basa en jornadas laborables de 8h pero se ha de tener en cuenta que la dedicación real ha sido de media jornada (entre cuatro y cinco horas). El tiempo dedicado a las pruebas de cada meta se incluye dentro del tiempo de la meta en concreto.

El proyecto global se divide en dos partes que se han desarrollado en paralelo, la del circuito electrónico para la iluminación y obtención de la velocidad, y la parte a la refiere el presente documento, la encargada del diseño de la aplicación móvil y conexión con el circuito electrónico. La planificación que se muestra a continuación es la relativa a la segunda parte del proyecto.

Meta	Descripción	Tecnología	Estimación (jornadas)
1	Inicio y configuracion de Raspberry	Raspberry	2
2	Conexión Bluetooth vía script Python	Raspberry	14
3	Creación app Android conexión Raspberry-Smartphone	App Android	15
4	Envio y recepcion de datos via Bluetooth	Raspberry + App Android	5
5	Creación script Python control de LEDs	Raspberry	1
6	Control LEDs vía App (Smartphone)	App Android	2
7	Obtencion de la velocidad y dibujo de un mapa (app)	App Android	8
8	Trazado ruta en mapa	App Android	13
9	Estudio de viabilidad: Lectura PWM por Raspberry	Raspberry	3

TABLA 2.1. PANIFICACIÓN

2.2. Presupuesto

En este apartado se hace un estimación de presupuesto necesario para llevar a cabo la parte del proyecto referente al presente documento, teniendo en cuenta los sueldos de los integrantes del equipo y los materiales y software necesarios para llevar a cabo las tareas.

Como se puede observar en la planificación realizada en Sección 2.1 del presente documento, el numero de jornadas laborales que se han de dedicar de un programador junior son 63. Teniendo en cuenta los impuestos y 12 sueldos, el coste de un programador

junior es de aproximadamente 78 €/jornada. En cuento al apoyo por parte del profesor se considerará el sueldo de un Project Leader, tras impuestos y con 12 pagas que el coste por jornada es aproximadamente 120 €/jornada con una dedicación de una jornada por cada 10 del programador.

Rol en el proyecto	**Coste (€/jornada)**	**Dedicación (jornadas)**	**Coste Total**
Programador Junior	78	63	4.914€
Project Leader	120	6.3	756€
		TOTAL	**5.670€**

TABLA 2.2. COSTE DEL PROGRAMADOR Y EL PROJECT LEADER

En cuanto a los materiales necesarios para llevar a cabo el proyecto se listan a continuación con su coste, todos los precios incluyen IVA y los que no son de nueva adquisición incluyen el precio tras amortización. También se incluyen todos los Sistemas Operativos (SO) y software necesarios.

Material	**Coste**
PC con Windows (1 año de uso)	350€
Smartphone Android gama Media-Alta (2 años de uso)	90€
Raspberry Pi 3 Model B+	45€
Protoboard	5€
Polimetro (1 año de uso)	2€
Cables Dupont M-F	2€
LEDs varios	1€
Resistencias varias	1€
Software y SO (Android Studio, Raspbian, Putty, ...)	0€
TOTAL	**496€**

TABLA 2.3. COSTE DEL MATERIAL, SOFTWARE Y SO

Incluyendo materiales, sistemas operativos, software y el personal el presupuesto total es de **6.166€**.

2.3. Entorno socio-económico

En este apartado se comentará el entorno actual en el que se implantará este proyecto a la vez que el impacto que se pretende y/o espera que tenga el proyecto sobre la sociedad, medio ambiente y economía.

2.3.1. Entorno e impacto Social

Este proyecto propone un sistema de iluminación que aportaría un factor seguridad para patinadores y ciclistas. La nueva normativa de circulación para la ciudad de Madrid limita el transito bicicletas y patinetes por zonas peatonales, esto les obliga a circular por las aceras junto a los automóviles y motocicletas. Frente a estos medios de transporte las bicicletas se encuentran en desventaja debido a su fragilidad y en caso de accidente tienen menos protecciones de las que tendría un usuario en un automóvil. El aumento de la visibilidad de los ciclistas de cara a los conductores de automóviles y motocicletas puede implicar una gran diferencia en cuanto al numero de accidentes de trafico.

Ademas este proyecto propone el uso de bicicletas y patinetes como medio de transporte, lo que supone un efecto beneficioso para el usuario por la actividad física que puede significar. Según [2][3] la actividad física es un medio de protección contra enfermedades como la hipertensión arterial o la obesidad ademas de ser un gran tratamiento para trastornos psicológicos como la depresión, ansiedad o el estrés.

Se pude concluir que este proyecto pretende hacer un aporte positivo a la sociedad, fomentado la actividad física a la par que aumentando la seguridad vial creando un método para la prevención de accidentes.

2.3.2. Entorno e impacto Económico

El impacto económico directo del sistema de iluminación propuesto se encuentra ligado al aumento del uso de vehículos alternativos en la ciudad de Madrid. Los cambios en la normativa de circulación de Madrid limitan el uso de los vehículos más contaminantes, esto está incentivando el mercado de transportes alternativos. El número de empresas de alquiler por minuto de coches, bicicletas y patinetes eléctricos se ha incrementado este ultimo año, un ejemplo de esto es el numero de empresas que ofrecen estos servicios siendo al menos cuatro[3] para alquileres de bicicletas y seis[4] para patinetes.

Estos medios de transporte alternativos se utilizan para movimientos dentro de la cuidad, normalmente para distancias cortas lo que hace poco prácticos los coches de alquiler ya que son más difíciles de encontrar, más caros y teniendo en cuenta los atascos en las carreteras más lentos. Esto unido a que pueden ser alquilados a partir de los 15 años hace que los patinetes y bicicletas abarquen un publico mas amplio y sean la opción mas accesible, rápida y barata.

El sistema de iluminación propuesto en el presente documento puede ser integrado en los vehículos sin necesidad de hacer modificaciones a los mismos lo que implica un coste mínimo para las empresas que quisieran integrarlo en sus flotas. Ademas disponer de este sistema es un incentivo ya que el sistema de iluminación aumenta la seguridad del usuario, lo que puede ser un factor decisivo para este a la hora de elegir entre una

[3]Empresas de alquiler de bicicletas en Madrid: BiciMAD, Donkey Republic, OBike, Ofo

[4]Empresas de alquiler de patinetes en Madrid: VOI, Wind, Lime, Bird, Acciona, Jump

empresa u otra. Pero el sistema además de acoplarse a bicicletas o patinetes de alquiler también puede integrarse en los vehículos de los usuarios finales, lo que incluye a otro gran mercado al que podría interesarle la adquisición de este sistema de iluminación, es más, la normativa obliga a los usuarios de patinetes sin motor a llevar luces de posición en caso de circular por zonas de visibilidad reducida.

Teniendo esto en cuenta se puede concluir que el principal mercado a explotar es el de bicicletas y patinetes de alquiler debido al numero de empresas y la capacidad económica de estas, aunque puede extenderse a los usuarios que quisieran adquirir el sistema para sus vehículos personales.

Actualmente el sistema se encuentra en desarrollo y el objetivo a corto plazo es la creación de un prototipo funcional para acoplarlo en un patinete y ponerlo a prueba. Pero a largo plazo el objetivo es la adaptación y perfeccionamiento del diseño para integrarlo en una empresa de alquiler de patinetes eléctricos.

2.3.3. Entorno e impacto Ambiental

El impacto ambiental directo del proyecto deriva del uso de los transportes alternativos a los que se acoplará el sistema de iluminación. Uno de los objetivos del proyecto es el incentivar el uso de bicicletas y patinetes frente a otros medios de transporte más contaminantes como pueden ser los automóviles o motocicletas.

A día de hoy las ciudades emiten mas del 60 % del dioxido de carbono siendo los edificios y el transporte los principales contribuidores[1] por lo que el uso de vehículos alternativos es una medida que ayudará a reducir las emisiones producidas por parte del trasporte. El método que se utiliza en este proyecto para fomentar el uso de medios de transporte alternativos se basa en hacer más segura la experiencia del usuario, esto se consigue aumentando su visibilidad del ambiente y respecto a otros usuarios de la vía.

De manera indirecta se debe tener en cuenta que el sistema de iluminación propuesto necesita de una alimentación eléctrica que implica un gasto energético. Según datos de [4] las emisiones directas producidas por el la generación de energía eléctrica por parte de medios de producción no renovables implica el 23,6 % de las emisiones directas de CO_2 de España.

Teniendo todo esto en cuenta se podría decir que el impacto medioambiental de este proyecto es medio, ya que a pesar de que se fomente el uso de medios de transporte alternativos se requiere de una alimentación energética que implica unas emisiones contaminantes a la atmósfera.

2.4. Marco regulador

Este proyecto surge debido a la Nueva Ordenanza de Movilidad para la ciudad de Madrid que regula la circulación de monopatines y patinetes, esta norma limita la velocidad de los patinadores a **10Km/h en zonas peatonales** y por carriles bici no protegidos deben circular con casco, elementos reflectantes y cuando la visibilidad sea reducida con **luces de posición**. La limitación de circular a la velocidad de un peatón también es mencionada en el «Artículo 121. Circulación por zonas peatonales. Excepciones.» en el Capitulo IV del «TÍTULO III. Otras normas de circulación» del Reglamento General de Circulación[5].

Las bicicletas son mas habituales de ver que los monopatines y patinetes que han estado surgido estos últimos años, por lo que tienen mayor numero de normativas que respetar. En el Reglamento General de Circulación anteriormente mencionado, concretamente en el «TÍTULO II. De la circulación de vehículos» del Capitulo X el «Artículo 98. Normas generales.» concreta que las bicicletas deben disponer de elementos reflectantes y precauciones necesarias para ser distinguidos a 150 metros cuando sea necesario alumbrado. Esto también se menciona en la Ley sobre Trafico[6] en el «Artículo 43. Uso obligatorio.» del «Capítulo II. Circulación de vehículos».

Y por ultimo en el Reglamento de Vehículos[7] en el «Artículo 22. Ciclos y bicicletas» del «TÍTULO III Ciclomotores, ciclos, vehículos de tracción animal y tranvías» y en la «LEY 43/1999, de 25 de noviembre, sobre adaptación de las normas de circulación a la práctica del ciclismo.» se especifica que se ha de utilizar iluminación en caso de circular de noche por túneles o en casos cuando la visibilidad sea baja como determinadas condiciones meteorológicas o ambientales.

En cuanto a la aplicación para poder acceder a los datos de ubicación el usuario ha de conceder el permiso como se especifica en la «Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales.», de este modo la aplicación podrá acceder y registrar los datos de ubicación.

[5]Real Decreto 1428/2003, de 21 noviembre, por el que se aprueba el Reglamento General de Circulación para la aplicación y desarrollo del texto articulado de la Ley sobre Tráfico, Circulación de Vehículos a Motor y Seguridad Vial, aprobado por el Real Decreto Legislativo 339/1990, de 2 de marzo de 1990.

[6]Real Decreto Legislativo 6/2015, de 30 de octubre, por el que se aprueba el texto refundido de la Ley sobre Tráfico, Circulación de Vehículos a Motor y Seguridad Vial.

[7]Real Decreto 2822/1998, de 23 de diciembre, por el que se aprueba el Reglamento General de Vehículos.

2.5. Listado de acrónimos

AC	Alternating Current
API	Application Programming Interface
BLE	Bluetooth Low Energy
DC	Direct Current
DGT	Dirección General de Trafico
EC	European Commission
EEUU	Estados Unidos
EGNOS	European Geostationary Navigation Overlay Service
ESA	European Space Agency
FPGA	Field Programmable Gate Array
GJU	Galileo Joint Undertaking
GLONASS	Global'naya Navigatsionnaya Sputnikovaya Sistema
GNSS	Global Navigation Satellite System
GPIO	General Purpose Input and Output
GPS	Global Positioning System
HDL	Hardware Description Language
HDMI	High-Definition Multimedia Interface
IDE	Integrated Development Environment
LED	Light Emitting Diode
MAC	Media Access Control
MFSAS	Multi-functional Satellite Augmentation System
MCC	Mission Control Centres
NLES	Navigation Land Earth Stations
PC	Personal Computer
PWM	Pulse-Width Modulation
RAM	Random Access Memory
RFCOMM	Radio Frequency Communications
RIMS	Ranging and Integrity Monitoring Stations
SDK	Software Development Kit
SIG	Special Interest Group
SIITR (I)	Sistema de Iluminación Inteligente para Transporte Recreativo (I)
SO	Sistema Operativo
UE	Unión Europea
USB	Universal Serial Bus
VMU	Vehículos de Movilidad Urbana
WAAS	Wide Area Augmentation System

2.6. Estructura del documento

El presente documento ha sido organizado del siguiente modo:

- En el Capítulo 3 se ha descrito el estado actual e historia de las lineas de investigación sobre las que se basa el presente proyecto. También se ha detallado el estado actual e historia de la tecnología Bluetooth.
- En el Capítulo 4 se explica cual ha sido la solución diseñada y se enumeran los componentes y lenguajes de programación seleccionados junto a una justificación de su selección.
- En el Capítulo 5 se detalla el desarrollo llevado a cabo, tanto de hardware como software. Además se añade un estudio realizado para evaluar la posibilidad de integrar el sistema diseñado con un proyecto que se esta desarrollando en paralelo.
- En el Capítulo 6 se evalúan los experimentos realizados con el fin de evaluar la precisión el funcionamiento del sistema final.
- En el Capítulo 7 se recopilan las conclusiones de los resultados experimentales y los trabajos futuros que se desarrollarán para la creación de un prototipo final totalmente funcional.
- También se han adjuntado el Anexo A y Anexo B que incluyen partes relevantes del código desarrollado y el Anexo C con resultados del estudio realizado en el Capítulo 5.

3. ESTADO DEL ARTE

En la siguiente sección se comentará el estado del arte de las diversas lineas de investigación y tecnologías que se han empleado durante el desarrollo del sistema planteado.

La solución propuesta de trata de un sistema de iluminación y navegación para transportes recreativos como patinetes y bicicletas, por ello se considera conveniente comentar es estado del arte de los Sistemas de Iluminación para la Seguridad Vial y de los Sistemas Globales de Navegación por Satélite (GNSS). En cuento a tecnologías la base del proyecto ha sido el Bluetooth, utilizado para conectar el Sistema de Iluminación con el de Navegación.

3.1. Sistemas de Iluminación para la Seguridad Vial

Los sistemas de iluminación[5] de los medios de transporte son una de las principales medidas para evitar los accidentes de tráfico. Esta demostrado que de la información que necesita un conductor para llevar a cabo correctamente su función el 90 % es obtenido mediante la vista y el 10 % restante se obtiene del oído y el equilibrio. Esto evidencia la importancia un un buen sistema de iluminación que permita una buena visibilidad para el conductor a la par que señalizar la posición del vehículo para poder llevar a cabo correctamente la función de «ver y ser vistos».

Por la noche, la capacidad visual del conductor se reduce un 80 % sobre la capacidad diurna lo que hace imprescindible una buena iluminación durante la noche. Las estadísticas elaboradas por la DGT demuestran que entre el atardecer y el amanecer se producen el 42 % de las victimas mortales a pesar de que el volumen de trafico sea mucho menor. La accidentalidad nocturna se puede explicar por diversos factores, como la relación entre noche y alcohol, o la somnolencia, pero esta demostrado que una buena visibilidad es un elemento imprescindible para la seguridad a la hora de circular de noche.

3.1.1. Evolución de los Sistemas de Iluminación

Se ha avanzado mucho en los sistemas de iluminación[5] desde la popularización de los automóviles ya que se hizo necesario algún tipo de iluminación para circular de noche. Así, a finales del siglo XIX fue como surgieron los primeros faroles con velas.

No fue hasta el siglo XX cuando se empezó a utilizar iluminación eléctrica principalmente para señalizar la posición del vehículo de cara al resto de usuarios de la vía. No se utilizaron para iluminar la vía porque se corría el riesgo de que se agotara la batería durante la marcha y de ser así no se podrían recargar sin hacer una parada lo que suponía un gran peligro si ocurría de noche.

Con la invención de la dinamo de alumbrado la industria de la iluminación para los

automóviles sufrió una revolución ya que a partir de entonces se podía generar energía eléctrica a partir del movimiento del propio vehículo. Ya no se corría el riesgo de que la iluminación se apagara durante la marcha por lo que la iluminación eléctrica comenzó a extenderse y a evolucionar según las necesidades que los vehículos iban planteando.

Una de las aportaciones que más influyo en la seguridad vial fue la de Florence Annie Bridgwood[6], precursora de los intermitentes y las luces de freno. Florence era una actriz de origen canadiense apasionada por los automóviles, a los cuales les dedicaba mucho tiempo, no solo conduciéndolos sino también buscando maneras de hacerlos mejores. Florence diseño un sistema por el cual cuando se pulsaba el freno de su automovil se desplegaba un cartel en la parte trasera del vehículo con la palabra «Stop», esto seria el origen de lo que hoy en día son las luces de freno. También ideo un sistema que, mediante el uso de un botón, se indicaba hacia que lado se giraría. A día de hoy estos diseños se han modificado para mejorarlos añadiéndoles luz, de esta manera son mas eficaces ante situaciones de baja visibilidad.

3.1.2. Iluminación en bicicletas, patinetes y monopatines

Hoy en día con la Nueva Ordenanza de Movilidad para la ciudad de Madrid las restricciones son mayores tanto para las bicicletas como para patinetes y monopatines por lo que se prevé una gran avance en las medidas de seguridad. Aunque los sistemas de iluminación para monopatines y patines son menos comunes, los de bicicletas llevan existiendo varios años. Esto es debido a que las bicicletas son mas comunes y llevan años circulando por las ciudades mientras los patinetes y monopatines se han dedicado, más que a ser medios de transporte, a ser actividades de ocio.

Las bicicletas, como vehículo, están obligadas a utilizar, entre otras medidas de seguridad, luces por la noche. La luz que se ubique en la parte delantera de la bicicleta debe ser blanca, y la que se coloque en la trasera deberá ser de color rojo. Ademas estas luces deben ser iluminación fija, no intermitente.

Historia de la iluminación para bicicletas

Cuando surgió la bicicleta los ciclistas sintieron la necesidad de añadirle algún tipo de iluminación[7] para poder circular de noche, y ante la situación empezaron a utilizar lamparas que tenían diferentes usos y no estaban adaptadas al ciclismo. Ante la situación surgió un nuevo tipo de luces específicamente diseñadas para ser acopladas a una bicicleta, estas lamparas funcionaban a base de petroleo y aceite.

Fig. 3.1. Lampara iluminación para bicicleta[7]

Al igual que los automóviles la electricidad no llegó hasta el siglo XX gracias al desarrollo de la primera dinamo. En uno de los modelos de lampara mas populares de la época el generador esta en contacto con la rueda, y al girar esta se genera la electricidad para iluminar la bombilla. Este diseño persiste en los sistemas de los que se dispone a día de hoy.

Fig. 3.2. Lampara con dinamo para bicicleta[7]

Actualidad de la iluminación para bicicletas

Aunque las lamparas y bombillas han cambiado mucho[8] la esencia sigue siendo la misma, un generador para crear energía a partir del movimiento de la bicicleta -antiguamente dinamos y a día de hoy pueden ser baterías, generadores magnéticos y también dinamos- y una bombilla.

Los generadores han cambiado bastante desde la primera lampara para bicicleta. Con las primeras lamparas para bicicletas los únicos medios de los que se disponía para generar electricidad eran las dinamos pero hoy en día se puede generar energía con varios tipos de

generadores. Lo mas utilizado en España son las luces con baterías o pilas ya que son los mas fáciles de instalar. Este tipo de iluminación suele basarse en pilas recargables debido al alto consumo del sistema, incluso hay sistemas que disponen de paneles solares para recargar la batería.

A pesar de los diversos sistemas disponibles las dinamos siguen siendo el método mas popular para dotar de iluminación las bicicletas. Las dinamos son el meto mas eficaz, ecológico y a la larga las mas económicas ya que para los sistemas de pilas se han de cambiar cada vez que pierden carga pero las dinamos una vez instaladas son un medio con bajo mantenimiento.

Fig. 3.3. Dinamo en un sistema de iluminación actual[8]

Al igual que los generadores han sufrido diversos cambios las bombillas utilizadas para la iluminación han pasado por el mismo proceso. Cuando se origino la primera lampara eléctrica para bicicletas solo existían las bombillas incandescentes por lo que eran estas las que se usaban.

Hoy en día las bombillas incandescentes han sido sustituidas por las lamparas LED[9]. Las lamparas LED tienen una vida útil unas 12,5 veces mas larga que una lampara incandescente y son mucho mas pequeñas en tamaño. Ademas las LED tienen un consumo muy bajo y se genera menos calor del que generaría una lampara incandescente. Todos estos motivos han hecho que la industria de la iluminación -al menos para la iluminación de bicicletas- haya tendido a utilizar este tipo de tecnología frente a las bombillas incandescentes.

Tendencias de futuro y proyectos

Aunque actualmente[8] son los menos comunes los generadores magnéticos están sufriendo un gran impulso gracias al uso de sistemas de iluminación LED. Este método se basa en colocar un imanes en los radios de la rueda -generalmente la trasera- y una

bobina en el chasis de la bicicleta de de manera que se genera una corriente eléctrica por la bobina con cada giro de la rueda.

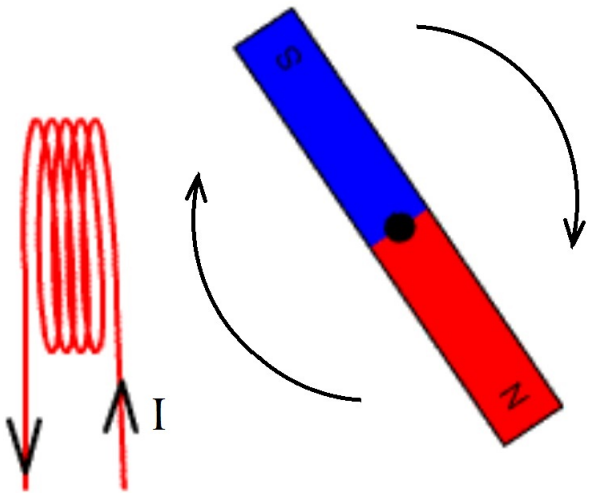


Fig. 3.4. Funcionamiento generadores magnéticos

Uno de los grandes retos para la iluminación de patinetes patines y bicicletas es conseguir almacenar la energía generada y conseguir tener baterías de gran capacidad con un pequeño volumen para poder alimentar los sistemas de iluminación que están siendo desarrollados. Uno de estos sistemas es el «light-bike»[10] que consiste en un láser que colocado en la parte baja del asiento de la bicicleta que emite un haz de luz indicando la distancia de seguridad entre la bicicleta y los coches. De esta manera se protege al ciclista dándole mas información al conductor de como adelantar correctamente.

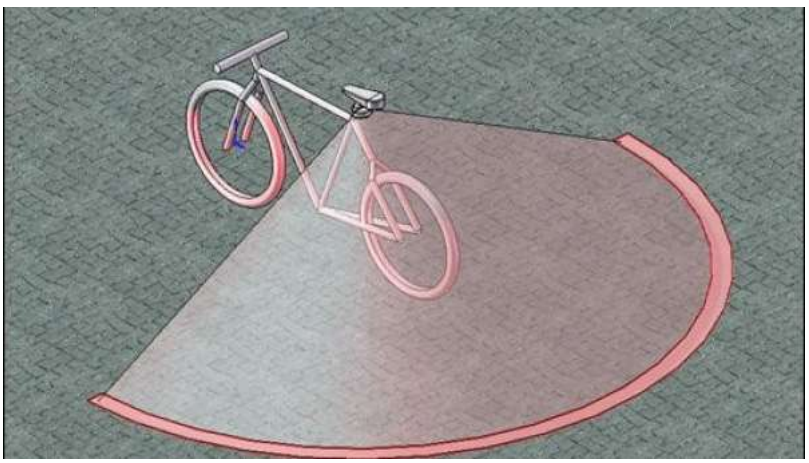


Fig. 3.5. Diagrama de fincionamiento de Light-bike[10]

También cabe destacar unos proyectos muy similares al planteado en el presente documento, el de Michael Standley[11] que propone un sistema de iluminación para monopatines o el de Jon P. Kertes[12] que propone uno para patinetes. Estos proyectos tienen el mismo objetivo principal, aumentar la seguridad facilitando la visibilidad del y para el

patinador. La principal diferencia con el presente proyecto es que este además ofrece una aplicación móvil con un sistema de navegación y un control inalámbrico del sistema de iluminación ofreciendo un producto final que pueda ser integrado en cualquiera de estos vehículos.

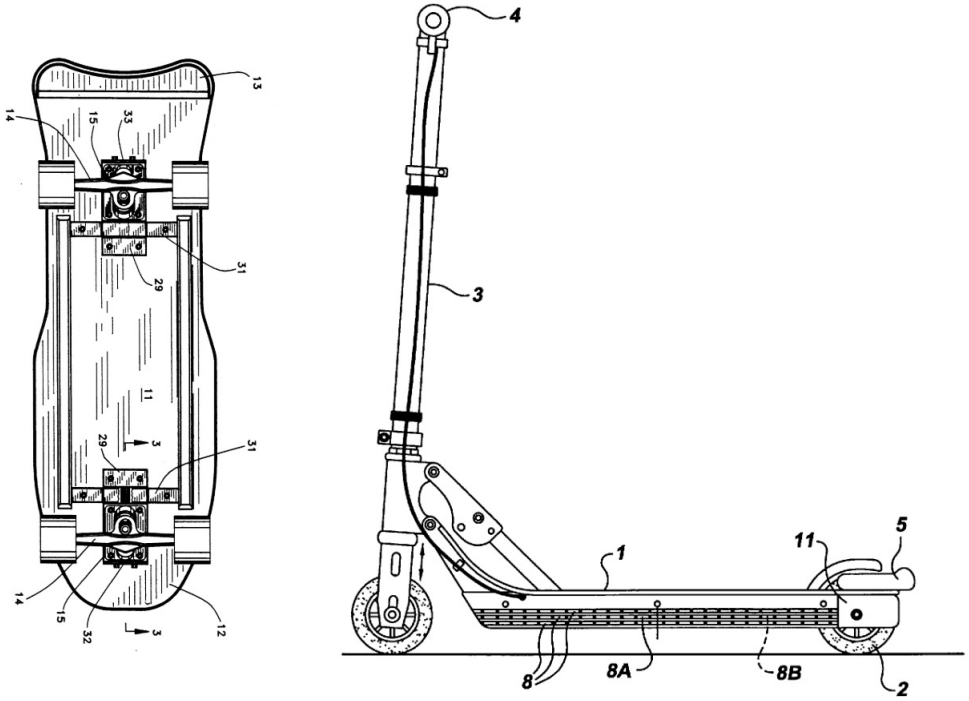


Fig. 3.6. Sistema de iluminación para monopatín[11] y patinete[12]

3.2. Sistema Global de Navegación por Satélite (GNSS)

Un Sistema Global de Navegación por Satélite, en adelante GNSS, es el conjunto de sistemas capaces de proporcionar información sobre posición y tiempo en cualquier lugar y momento. Los primeros GNSS fueron el GPS (Sección 3.2.2) y el GLONASS, actualmente la mayoría de dispositivos móviles incluyen antenas GPS y GLONASS, y ya hay modelos más recientes que también incluyen Galileo (Sección 3.2.4), pero GPS sigue siendo el principal mientras los demás GNSS se utilizan como apoyo para mejorar precisión.

3.2.1. ¿Como funcionan los GNSS?

Los GNSS son sistemas formados por un grupo de satélites en órbita y una serie de bases en la Tierra desde donde se controlan. Los satélites se encuentran constantemente emitiendo señales hacia la Tierra, cuando a un receptor le llega alguna de estas señales calcula el tiempo que ha tardado en llegar desde el satélite, de esta manera puede medir la distancia entre ambos.

Ya que la posición de los satélites es conocida lo único que el receptor necesitará es

saber es la distancia entre si mismo y cuatro satélites, tres para posicionarse y el cuarto para eliminar los errores de sincronismo[13].

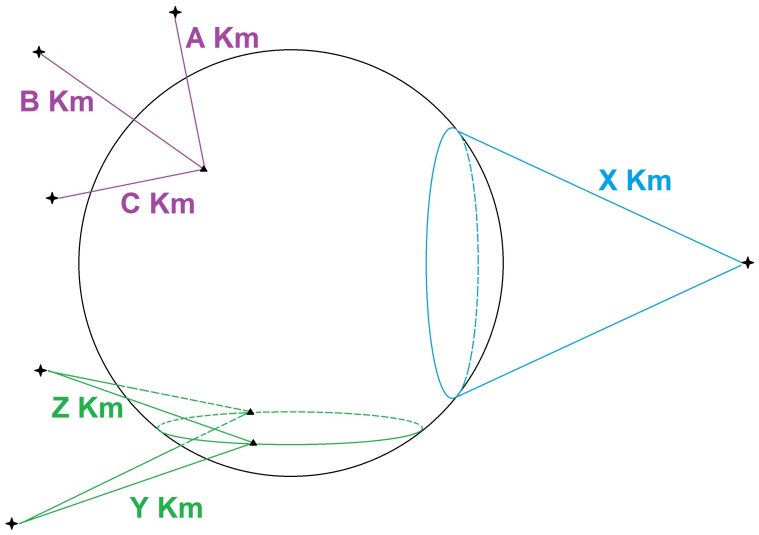


Fig. 3.7. Método de triangulación para el posicionamiento

Son necesarios tres para el posicionamiento ya que de usarse solo uno no se tendría suficiente información para ubicar el receptor en un solo punto de la Tierra. De usarse dos satélites, se podría ubicar al receptor en dos puntos diferentes y no se sabría con certeza cual de los dos puntos es el correcto, añadiendo un tercer satélite se puede posicionar inequívocamente el punto sobre la superficie de la Tierra en el que se encuentra el receptor.

Todos los GNSS están formados por tres segmentos básicos[14]:

- Segmento Espacio, formado por los satélites en órbita.
- Segmento Control, formado por las estaciones encargadas de mantener en órbita los satélites y controlar que funcionen de manera precisa.
- Segmento usuario, formado por los dispositivos receptores.

Segmento Espacio

El segmento Espacio[15] es el formado por todos los satélites en órbita con la Tierra, tanto los satélites de navegación como los de comunicación.

Cualquier GNSS debe contar con suficientes satélites de navegación como para poder ofrecer una cobertura global en cualquier momento. Ademas se debe garantizar el servicio a pesar de que, por algún motivo, un satélite dejase de prestar servicio, esto se consigue teniendo mas satélites de los estrictamente necesarios. Ademas los satélites de navegación deben estar ubicados en diferentes órbitas de manera que se cubra toda la superficie terrestre.

Fig. 3.8. Satélites GNSS[15]

Los satélites de comunicación son satélites con la función de comunicarse con el segmento control para transmitir correcciones, aumentando la precisión de los satélites de navegación, estos son los conocidos como sistemas de aumento. Los satélites de comunicación son particulares de cada país conformando sus propios sistemas de aumento, en Japón y Australia se utiliza MFSAS, en EEUU el WAAS, en Europa el EGNOS, etc.

Fig. 3.9. Sistemas de Aumento[15]

Segmento Control

El segmento Control[15] es formado por la estaciones que recogen los datos del estado de los satélites. Su función es la de verificar el correcto funcionamiento del segmento Espacio desde la Tierra.

El Segmento control se encarga de aplicar correcciones de posición orbital y temporal de los satélites mediante el envió de datos de sincronización de relojes atómicos y posicionamiento orbital. Estos datos se pueden tratar de dos modos: enviándolos de la estación de

control a los satélites de navegación, o bien enviándolos a los satélites de comunicación.

El primer método es el empleado por las estaciones de control de EEUU y la Federación rusa con los satélites GPS y GLONASS respectivamente, este método permite tratar activamente los errores.

El segundo método es el utilizado, por ejemplo, por las estaciones de control europeas con los satélites de EGNOS, este método no influye activamente sobre los satélites de navegación sino que se utiliza para tener en cuenta los errores a la hora del calculo de la posición.

Segmento Usuario

El Segmento Usuario es el compuesto por los dispositivos receptores. Los dispositivos receptores son capaces de recibir las señales de los satélites e interpretarlas para poder obtener la posición y el tiempo en el que se encuentran.

Estos dispositivos están compuestos por dos partes, la antena y el receptor. La antena receptora, como su nombre indica, es la encargada de recibir las señales de los satélites. El receptor es el encargado de adaptar la señal recibida para poder procesarla y calcular la posición sobre la superficie de la Tierra, el receptor también incluye un reloj y una pantalla para mostrar la información.

Para el calculo de la posición, el receptor tiene tres tareas principales: «Satellite Manager», es la función de almacenaje del estado de los satélites y los datos necesarios para hacer los cálculos; «Select Satellite» que consiste en seleccionar los cuatro satélites con mejor posición para el calculo y «SV Position Velocity Acceleration» que se encarga de calcular la velocidad y posición de los satélites.

Los receptores de hoy en día son capaces de captar señales de mas de 15 satélites a la vez e interpretar señales de varios GNSS o sistemas de aumento. Esto significa que con un solo receptor se puede obtener información de un satélite GPS, uno GLONASS, uno Galileo y un satélite de comunicación como los de EGNOSS.

3.2.2. GPS

El Sistema de posicionamiento Global, en adelante GPS, es un sistema de localización capaz de proporcionar información en tiempo real sobre la posición, hora y velocidad. Actualmente el GPS es el único GNSS totalmente operativo.

El sistema comenzó su desarrollo en 1973 gracias al Departamento de Defensa de Estados Unidos (EEUU), pero no fue totalmente funcional hasta el año 1995 cuando se lanzaron la totalidad de los satélites de navegación y se encontraban operativas la estaciones de control.

En 1972 se realizaron una serie de pruebas para comprobar la precisión del sistema, estas pruebas le concedieron una fiabilidad de entre 1 y 15 metros. Ante estos resultados

el Departamento de Defensa de los Estados Unidos se alarmó con la posibilidad de que el sistema pudiera ser utilizado por sus enemigos, así se origino la disponibilidad selectiva.

La disponibilidad selectiva es un error de precisión introducido voluntariamente para reducir la precisión de los sistemas GPS para los receptores civiles. Esta técnica reducía la precisión vertical a 156 metros y la horizontal entre 15 y 100 metros. Finalmente el 2 de mayo del año 2000 la disponibilidad selectiva fue eliminada y dio paso a la tecnología GPS mas precisa que se conoce a día de hoy[16].

A pesar de que a disponibilidad selectiva asido eliminada los EEUU siguen manteniendo ciertas restricciones como limitar que los receptores GPS no sean operativos para velocidades mayores a 515 m/s (metros por segundo) y alturas superiores a 18 Km (Kilómetros).

El GPS es un tipo de GNSS, por lo que esta formado de tres segmentos, el Segmento Espacio, el Segmento Control y el Segmento Usuario.

Segmento Espacio

El GPS originalmente fue diseñado para funcionar con 24 satélites de navegación. Con el paso del tiempo se han ido añadiendo satélites para mejorar la precisión, y actualmente el sistema cuenta con 30 satélites en órbita.

Fig. 3.10. Distribución original de los 24 satélites GPS[15]

A fecha de hoy el GPS es el único GNSS totalmente funcional y el único capaz de garantizar un mínimo de 5 satélites disponibles en cualquier lugar del mundo en cualquier momento.

Segmento Control

El Segmento Control esta formado por tres Estaciones de Control Maestras (MCS) y varias estaciones de control. Las estaciones de control se encargan de recopilar la información de los satélites y enviarla a las MCS que serán las que apliquen las correcciones necesarias.

En cuanto a las Estaciones de Control Maestras la Estación de Control principal esta ubicada en Colorado en la base Falcon del Ejercito del Aire de EEUU, las otras dos son estaciones de reserva y están ubicadas en California y Maryland.

El resto de estaciones de control están ubicadas en Hawaii, Kwajalein, Ascension Island, Diego Garcia y Colorado Springs. En concreto Diego Garcia, Kwajalein y Ascension Island son estaciones capaces de comunicarse con los satélites.

Fig. 3.11. Segmento Control GPS[15]

3.2.3. EGNOS

El GPS y el GLONASS son sistemas de diseño militar por lo que los civiles no tienen acceso al segmento control, es gestionado por los departamentos de defensa de EEUU y la Federación Rusa respectivamente, este es uno de los principales motivos por los que la ESA (Agencia Espacial Europea) desarrolló el sistema EGNOS[15]. Otro de los motivos principales por los que se diseño EGNOS fue a la aparición de la aviación para civiles. La precisión del GPS y del GLONASS no eran suficiente para la aviación civil y se requería una mejora.

Sin los satélites EGNOS los errores de posición rondan los tres metros horizontalmente y cinco verticalmente, el satélite pretende reducir este error para mejorar, especialmente, la aviación civil, pero también para travesías marítimas y terrestres.

Adicionalmente EGNOS proporciona un referencia temporal estable con una preci-

sión de nanosegundos con aplicaciones como sincronización de redes para teléfonos móviles, sincronización de nodos de Internet, sincronización de redes eléctricas, etc. Además de proporcionar una amplia gama de nuevos servicios al combinar navegación por satélite con servicios móviles.

El segmento Espacio de EGNOS esta formado por tres satélites Inmarsat-3 AOR-E, Inmarsat-3 IOR y el satélite Artemis de la ESA. El área que cubrirá EGNOS será aproximadamente el mismo que el de la Conferencia Europea de Aviación Civil:

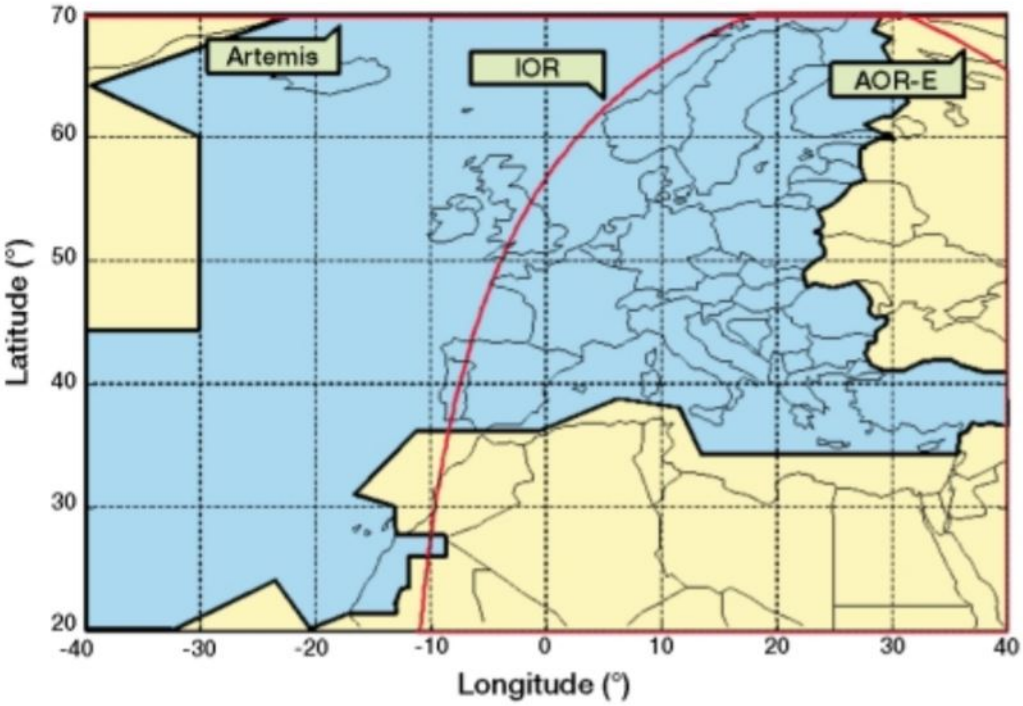


Fig. 3.12. Área cubierta por EGNOS[17]

El segmento Control de EGNOS esta formado por 34 estaciones RIMS, encargadas de supervisar los satélites; cuatro estaciones MCC, encargadas de procesar los datos de las RIMS y generar los datos de corrección y tres estaciones NLES encargadas de enviar los datos a los satélites.

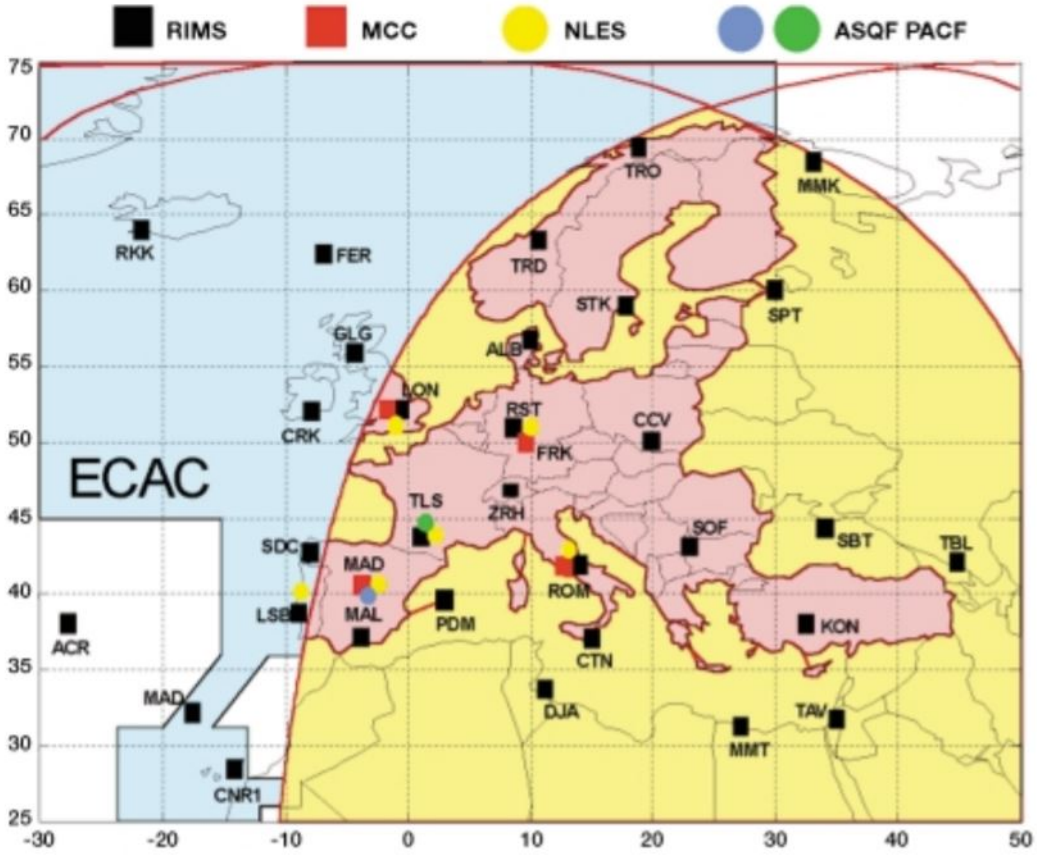


Fig. 3.13. Segmento Control EGNOS[17]

3.2.4. Lineas futuras: Galileo

Galileo[17] es un nuevo GNSS creado por la UE, que permitirá la independencia del GPS. Este sistema esta siendo desarrollado, no solo para tener una alternativa funcional al GPS, sino para que también pueda trabajar junto a él y al GLONASS para proporcionar un servicio más preciso.

La principal diferencia entre Galileo y GPS y GLONASS es que estos dos últimos son de desarrollo y mantenimiento militar lo que restringe el acceso de los civiles, en cambio Galileo es un proyecto llevado a cabo por la Comisión Europea (EC) junto a la Agencia Espacial Europea (ESA) con participación 27 países y multitud de empresas de la UE.

Desde 1999 hasta 2002 Galileo estuvo en fase de definición, esta fase consistió en la formación de alianzas orquestadas por la CE y la ESA. No fue hasta 2002 cuando empezó la fase de desarrollo. La CE y la ESA crearon la Galileo Joint Undertaking (GJU) que se encargaría de llevar a cabo el proyecto. En 2007 GNSS Supervisor Authority tomo las riendas del proyecto y lo dirigirá hasta que Galileo sea totalmente operativo.

En 2016[18] Galileo comenzó a operar con 18 satélites, el objetivo es tener en el Segmento Espacio 30 satélites de los cuales tres serán de repuesto. Galileo aún no es completamente funcional ya que no cubre totalmente la superficie terrestre y no puede responder a toda horas en cualquier lugar del mundo, pero la previsión es tenerlo total-

mente operativo para el año 2022.

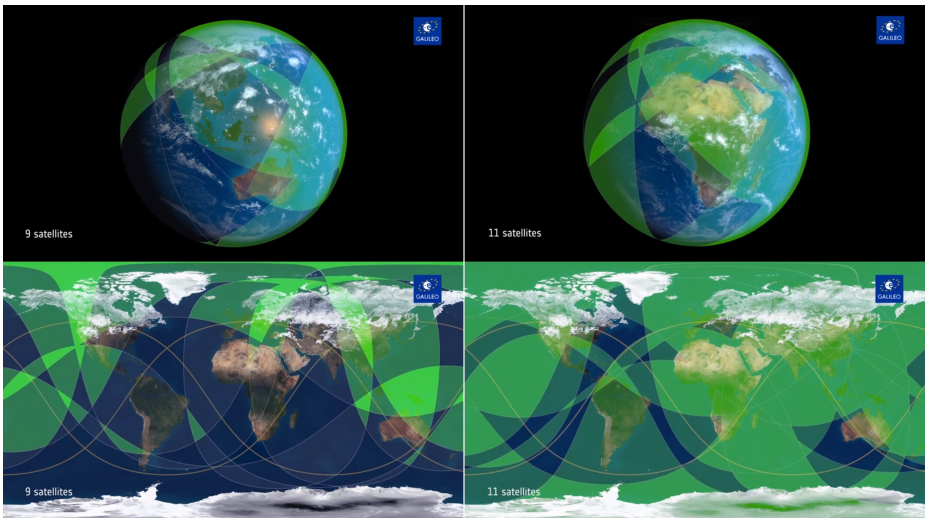


Fig. 3.14. Area puntual cubierta por 9 y 11 satélites[18]

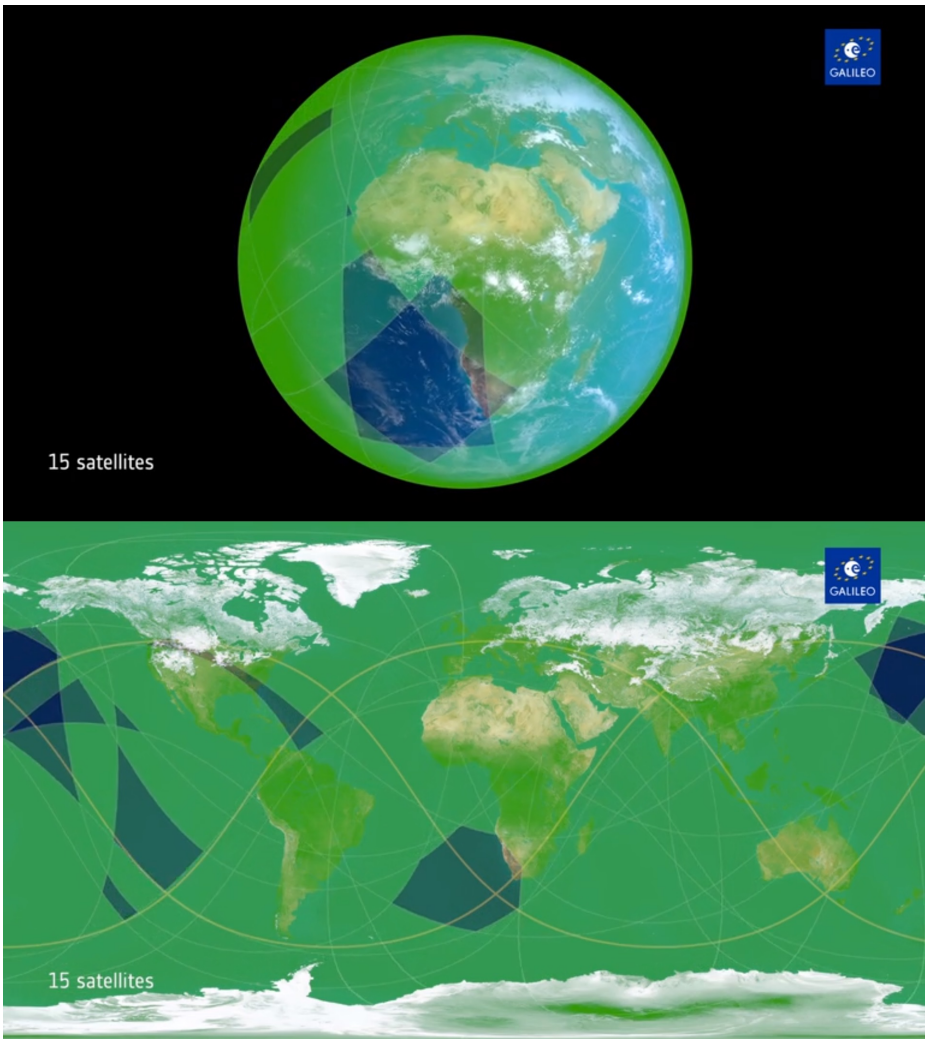


Fig. 3.15. Area puntual cubierta por 15 satelites[18]

En cuanto al Segmento Control seguirá una estructura similar a la del GPS, habrá dos tipos de estaciones, las estaciones de control que se encargan de recopilar la información de los satélites y enviarla a las estaciones maestras que serán las que apliquen las correcciones necesarias.

Habrá cinco estaciones maestras llamadas Galileo Control Center (GCC) situadas en la UE y 30 estaciones de recopilación de datos llamadas Galileo Sensor Stations (GSS) que se ubicarán por todo el mundo.

Fig. 3.16. Segmento Control Galileo[8]

3.3. Tecnologías: Bluetooth

Para este proyecto se ha optado por utilizar Bluetooth para la conexión entre el circuito electrónico y el teléfono móvil. Entre otros motivos se ha elegido esta tecnología porque a día de hoy es de las más comunes en los smartphones. Ya sean dispositivos con Android, IOS, Windows Phone o cual quier otro sistema operativo, todos disponen de chips Bluetooth en su hardware.

A continuación se entrará algo mas en detalle sobre los orígenes y el funcionamiento de esta tecnología.

3.3.1. Origen del Bluetooth

El Bluetooth (en adelante BT) tiene su origen en 1994 cuando la compañía Ericsson Mobile Communications inicio un estudio para implementar una tecnología que permitiera la conexión entre sus teléfonos móviles y sus accesorios. Para que la tecnología BT

[8] «The reference for Global Navigation Satellite Systems» *ESA Navipedia* `https://gssc.esa.int/navipedia/index.php/File:Galileo_s_Global_Ground_Segment.jpg` (Acceso: 19-05-2019)

fuera un éxito se necesitaba que hubiera una gran cantidad de dispositivos que la utilizaran, por ello a principios de 1998 Ericsson se unió a Intel, IBM, Nokia y Toshiba para formar el Grupo de Interés Especial (SIG)[19]. Con el paso del tiempo el SIG fue aumentando a sus mienbros y lo que empezó siendo un grupo de 5 compañías en 2017 ya contaba con mas de 30000.

Finalmente en 1999 el SIG lanzo la primera versión de BT que teóricamente, permitía una velocidad de hasta 1 mega-bit por segundo (Mbps) a una distancia menor de 10 metros. Para los consumidores, e incluso entre los círculos tecnológicos el BT era algo difícil de comprender, por eso no se popularizo hasta el año 2000 cuando se pusieron a la venta grandes cantidades de dispositivos con BT como teléfonos móviles, auriculares, ordenadores...[20]

El BT sigue siendo popular gracias a que ha seguido actualizándose con el paso del tiempo, mejorando su velocidad de transferencia de datos y manteniendo un bajo consumo. Sobre todo el factor de bajo consumo es el que ha dado origen al BT Low-Energy (BLE).

3.3.2. ¿Que es y como funciona?

El BT es un protocolo de comunicaciones inalámbrico destinado a comunicar dispositivos a corto alcance con bajo consumo. La comunicación entre dispositivos se realiza mediante microondas, estos no necesitan una visión directa entre ellos gracias a que las microondas son capaces de atravesar paredes, madera, tejidos...

A pesar de poder atravesar varios tipos de superficies el BT tiene la limitación de la distancia máxima entre dispositivos. La distancia máxima a la que se puede establecer una conexión depende de varios factores como la potencia de emisión, la sensibilidad del receptor, etc.

El BT trabaja en entre 2,4GHz y 2,485GHz, aunque en la practica el rango de trabajo es de 2,402GHz hasta 2,480GHz, esta banda de frecuencia se puede utilizar sin necesidad de licencia y es común a todo el mundo[21].

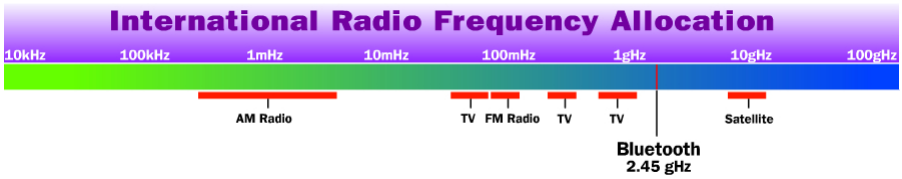


Fig. 3.17. Banda de trabajo del Bletooth[22]

Una técnica para evitar interferencias que utilizan los dispositivos BT es mandar señales muy débiles, de 1mW (un teléfono puede llegar a transmitir una señal de hasta 3W). Esta baja potencia limita el alcance del BT a aproximadamente 10m[22].

Existen 3 clases de dispositivos BT -las tres compatibles entre si- diferenciados por

la potencia a la que pueden emitir, ver Tabla 3.1. La mayoría de dispositivos de consumo como teléfonos móviles, PCs y auriculares son de clase 2.

	Potencia máx. emisión	Potencia máx. emisión	Alcance (Aproximado)
Clase 1	100 mW	20 dBm	100m
Clase 2	2.5 mW	4 dBm	10m
Clase 3	1 mW	0 dBm	1m

TABLA 3.1. CLASIFICACIÓN EQUIPOS BLUETOOTH[21]

La banda de frecuencias en la que trabaja el BT se divide en 79 sub-bandas o canales distintos, en España solo se dispone de 23 canales. Cada canal tiene un ancho de banda de 1MHz por el que se puede enviar y recibir datos. Para enviar o recibir datos un dispositivo salta desde un canal a otro con una frecuencia de 1600 veces por segundo hasta que encuentra un canal libre, de esta manera se evitan interferencias entre varios dispositivos que estén usando BT en la misma zona y al mismo tiempo[21][22].

Todo los dispositivos disponen de una dirección MAC (Media Access Control), la dirección MAC es un numero binario usado para identificar inequívocamente a un dispositivo. Esta dirección es grabada en el hardware durante su fabricación y esta diseñada para no poder ser modificada. La dirección MAC se utiliza para varios tipos de redes como el Wifi o el Ethernet[23], un escaner BT es capaz de leer la dirección MAC de los dispositivos que emiten en la banda de frecuencias del BT, de esta manera se identifica el dispositivo para poder establecer conexión.

4. DISEÑO DE LA SOLUCIÓN

En esta sección se especificará el diseño planteado para la solución descrita, concretando la estructura de la solución, la relación entre las capas que la componen y concretando cuales son los componentes seleccionados.

4.1. Funcionamiento del sistema

En Sección 1.2 se ha explicado la el funcionamiento del sistema en lineas generales, en esta sección del proyecto se entrará en detalle en las tecnologías utilizadas y en las herramientas ademas de sus funciones dentro del proyecto.

Para la conexión entre el circuito de iluminación y el dispositivo móvil que ejercerá la función de Sistema de Navegación, se utiliza Bluetooth y una Raspberry Pi. La Raspberry Pi es el medio de procesado de información, es decir, la Raspberry será capaz de interpretar las señales recibidas y comunicarlas vía Bluetooth al dispositivo móvil que mediante una interfaz gráfica mostrará la información al usuario final.

Cuando el proyecto este totalmente finalizado el Sistema de Iluminación diseñado en el proyecto SIITR (I) obtendrá la velocidad del patinador mediante un láser y enviará una señal a la Raspberry Pi, ademas este sistema se encargará de la iluminación del ambiente y de las luces de posición del patinador. La Raspberry procesará la señal recibida por parte del SIITR (I) y evaluará si se ha superado el limite de velocidad permitido. Una vez hecho esto la Raspberry mandará otra señal diferente al SIITR (I) y será este el que se encargue de iluminar un piloto de aviso. La Raspberry también mandará el dato de velocidad a la App vía Bluetooth para mostrar por pantalla la velocidad actual y la aplicación Android vía GPS obtendrá la posición del patinador y trazará la ruta.

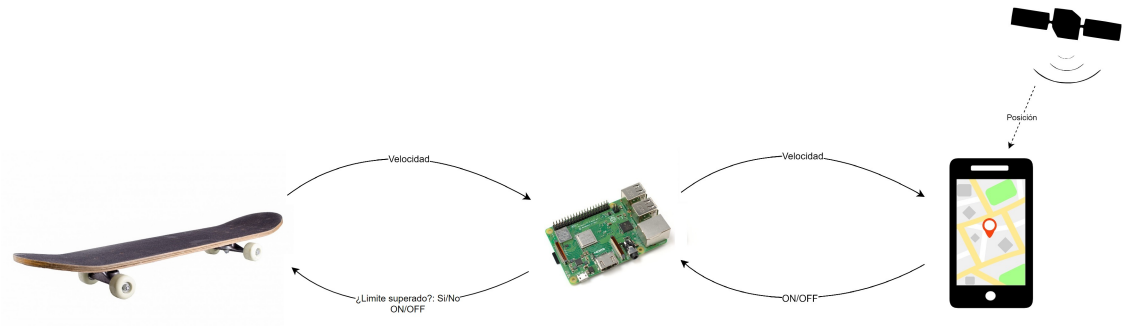


Fig. 4.1. Sistema diseñado y objetivo final

Debido a que la progresión de ambos proyectos ha sido diferente, para el presente documento se presenta un diseño distinto. En la solución descrita en este proyecto la

función de obtención de velocidad recae sobre la aplicación Android, que mediante GPS obtiene la velocidad del patinador. La misma aplicación evalúa si el patinador circula a una velocidad adecuada para ir por acera y manda un dato vía Bluetooth a la Raspberry Pi, esta ilumina un piloto verde o rojo. El piloto verde indica que la velocidad del patinador es la adecuada para circular por la acera junto a los peatones, en caso de iluminarse el piloto rojo el patinador debería disminuir su velocidad hasta que se apagara y se volviera a encender el piloto verde. En este desarrollo además se ha utilizado un LED azul para simular la señal de encendido y apagado del sistema de iluminación completo.

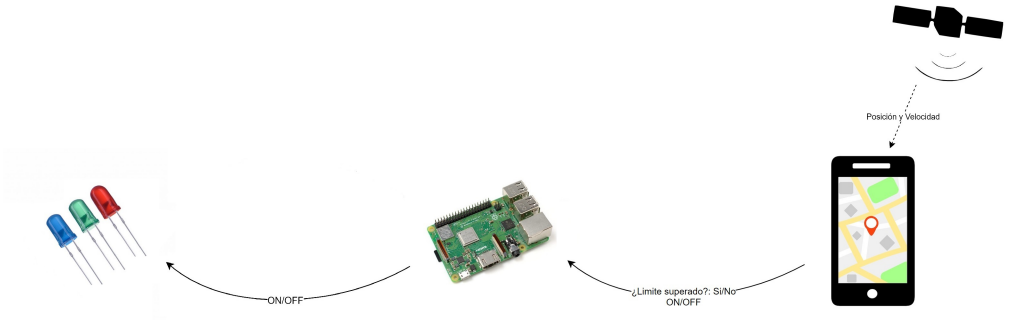


Fig. 4.2. Sistema desarrollado

4.2. Herramientas y criterios de selección

En este apartado se detallarán los motivos por los que se han elegido las herramientas seleccionadas y la función que tienen dentro del proyecto. Para la solución diseñada se ha optado por utilizar una Raspberry Pi para la comunicación el circuito electrónico y la aplicación móvil y se ha seleccionado un dispositivo con Android como sistema operativo para el diseño de la aplicación.

4.2.1. Dispositivo móvil

Para la elección de un dispositivo sobre el que desarrollar la solución los SO más accesibles son Android, iOS o Windows Phone. Los **requisitos mínimos** que debe cumplir el dispositivo son los siguientes: el dispositivo seleccionado debe disponer como mínimo de una antena receptora de señales GPS y disponer de Bluetooth.

Algunas ventajas, que a pesar de no ser necesarias pueden dar mejores resultados, es que el dispositivo que se seleccione sea compatible, ademas de con GPS, con GLONASS y Galileo y tener la versión Bluetooth similar o mayor a la de la opción seleccionada para el middleware. Teniendo esto en cuenta se analiza cual de las opciones de SO es la más adecuada.

Android

Android[24] es un SO basado en Unix diseñado para dispositivos móviles. Android fue originalmente diseñado para cámaras fotográficas, en 2005 Google compró Android y lo adapto para ser utilizado por dispositivos móviles. No fue hasta 2007 cuando Android fue finalmente liberado y se convirtió en un SO de código abierto.

Al diseñar Android los desarrolladores se fijaron en las características de los teléfonos móviles que creyeron que no cambiarían con el paso del tiempo como las pantallas táctiles o los botones de encendido. Un gran ventaja que tiene Android es que fue diseñado para poder ser instalado en cualquier tipo de móvil, no fue diseñado exclusivamente para un hardware por lo que no hace asunciones sobre, por ejemplo, tamaños de pantalla o tipos de procesador.

Para los desarrolladores Android proporciona todas las herramientas necesarias y acceso a todo el código de Android por lo que pueden comprender su funcionamiento y desarrollar aplicaciones más eficientes. Además Google proporciona Google Play, plataforma oficial de Android para la distribución de aplicaciones. Se pueden subir aplicaciones a Google Play de manera gratuita para los desarrolladores y sin largos procesos de verificación además de poder obtener beneficios de estas. Otra gran ventaja de Android es su popularidad, Android es el SO mas utilizado por lo que tiene un gran mercado a explotar.

iOS

La propiedad de iOS[25] pertenece a la compañía estadounidense Apple. IOS fue inicialmente desarrollado para iPhone pero actualmente se usa también para iPad, iPod y Apple TV, hasta fecha de hoy este SO solo eta disponible para productos fabricados por Apple y es un SO propietario. IOS fue lanzado en 2007 y ese mismo año se anunciaron que se lanzaría un SDK que permitiría desarrollo de aplicaciones de terceros. No fue hasta 2008 cuando finalmente el SDK fue liberado.

Windows Phone y Windows 10 Mobile

Al igual que iOS Windows Phone es un SO propietario, esta vez desarrollado por Microsoft. Windows Phone fue lanzado en 2010 diseñado exclusivamente para teléfonos móviles lo que derivó en su discontinuidad en 2015 para dar paso a Windows 10 Mobile que se puede usar, ademas de en teléfonos móviles, en tabletas y PCs.

A diferencia de iOS Windows 10 Mobile puede ser instalado en variedad de dispositivos siempre y cunado cumplan las condiciones de hardware. Microsoft pone en 2011 a disposición de los desarrolladores un SDK.

Opción seleccionada

Las facilidades que proporciona Android para los desarrolladores han sido un factor decisivo a la hora de elección de un SO sobre el que desarrollar la aplicación. Android es el SO más usado del mundo, más incluso que Windows para PC, por lo que una aplicación en Android abarca a un publico mayor que una aplicación para iOS o Windows Phone. Además Android es el único que es de código abierto.

El dispositivo seleccionado para el desarrollo es un smartphone de la marca Huawei, en concreto el Honor 7 con Android 6.0, Bluetooth 4.1 y antenas GPS y GLONASS.

4.2.2. Middleware

La organización que se ha seguido en este proyecto a sido algo diferente, normalmente se pasa por una etapa de definición donde se concretan las funcionalidades que se pretenden obtener y qué tecnologías y herramientas se emplearán para llevarla a cabo. En cambio para el presente proyecto, la metodología utilizada ha sido algo diferente, en la fase de definición se decidió concretar solamente los objetivos básicos del proyecto, dejando abiertas a modificaciones las funcionalidades más especificas.

Este tipo de organización permite una mayor flexibilidad al producto final pero obliga a emplear herramientas muy versátiles, que tengan diversos usos. Esto es debido a que de querer añadir un funcionalidad nueva no contemplada anteriormente habiendo seleccionado una herramienta especializada, la herramienta utilizada seria un limitante y las únicas opciones ante la nueva idea serian o descartarla o reimplementar el diseño sobre una plataforma nueva que si permitiera el desarrollo de la idea.

Para los **requisitos mínimos** se requería un dispositivo programable, que dispusiera de Bluetooth y que tuviera entradas y salidas digitales además de que la curva de aprendizaje fuera relativamente sencilla.

Arduino

Arduino es un microcontrolador integrado en una placa diseñada con objetivo educativo, lo que hace que su curva de aprendizaje sea muy sencilla. Arduino es la opción más sencilla para diseño de prototipos, es capaz de leer y escribir señales digitales y analógicas. El inconveniente de Arduino es que no es capaz de llevar a cabo tareas en paralelo.

Arduino llega de fabrica con Arduino IDE y no requiere de ningún periférico para comenzar a ser usada, con un PC y el IDE se puede empezar a trabajar sobre la placa. El IDE de Arduino usa una simplificación de C++ como lenguaje de programación por lo que resulta mas sencillo de aprender.

FPGA

Un FPGA[26] (Field Programmable Gate Array) es un dispositivo formado por bloques unidos por una serie de conexiones programables. Al programar un FPGA se modifica una matriz de conexiones, cada bloque programable esta construido de manera que puedan cambiar su función por lo que al programar un FPGA lo que se esta haciendo es construir un circuito electrónico real.

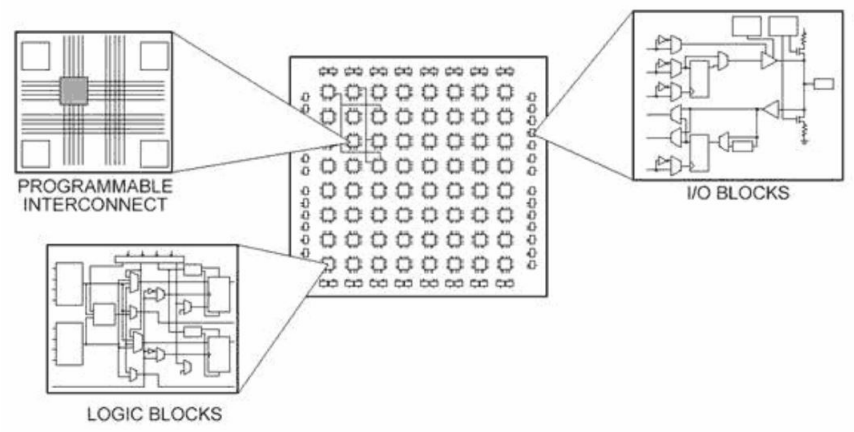


Fig. 4.3. Matriz de un FPGA[26]

Los FPGA no se programan con los lenguajes convencionales sino que se usan lenguajes descriptivos denominados HDL (Hardware Description Language). Este tipo de lenguajes tienen una curva de aprendizaje compleja, además de las opciones seleccionada es la menos económica y las herramientas de desarrollo suelen ser propietarias y especificas para cada fabricante y no suelen ser gratuitas.

Raspberry Pi

Una Raspberry Pi es un ordenador del tamaño de una tarjeta. Para comenzar a usar una Raspberry es necesario, como mínimo una tarjeta microSD y un PC. Una Raspberry es una placa que consiste de su propia memoria dedicada, procesador y SO basado en Linux.

La Raspberry Pi es una herramienta potente y capaz de realizar tareas que requieran gran capacidad de procesado, también es capaz de realizar tareas en paralelo y la placa está adaptada para acoplar todo tipo de periféricos. Además la curva de aprendizaje de la Raspberry es relativamente sencilla.

La Raspberry ademas es programable en cualquier lenguaje posible, por lo que se puede seleccionar el que mas se adapte a la aplicación a desarrollar. Todos estos motivos hacen que una Raspberry sea el medio ideal para prototipado de proyectos de media y gran envergadura.

Opción seleccionada

En este proyecto el sistema que se seleccione para el middleware debe tener la posibilidad de ejecutar tareas en paralelo, cosa que un Arduino no puede hacer. Aunque en el desarrollo presentado en este documento no es necesario, en el diseño planteado es una necesidad real. En la Figura 4.1 se puede observar que middleware debe ser capaz es estar «escuchando» al dispositivo móvil vía Bluetooth a la vez que lee el pin de entrada que le proporciona la información de la velocidad, esto no se podría hacer con un Arduino como middleware.

De las dos opciones restantes un FPGA es la mas cara y mas compleja por lo que finalmente se ha optado por una Raspberry Pi. Una Raspberry Pi es más potente de lo necesario, lo suficientemente versátil y la curva de aprendiza es asumible.

Finalmente la Raspberry Pi seleccionada es una Raspberry Pi 3 Model b+. El modelo seleccionado cuenta con un procesador Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz, 1GB RAM, Bluetooth 4.2, BLE, Wi-Fi, HDMI, cuatro puertos USB y 40 pines GPIO.

4.2.3. Lenguaje de programación

Come se comentó en Sección 4.2.1 y en Sección 4.2.2 se ha seleccionado una Raspberry Pi para el middleware y un dispositivo móvil con Android como SO. La selección de Raspberry Pi permite la elección de un lenguaje de programación, las opciones que mas viables fueron Java y Python ya que son los mas populares, con la comunidad mas amplia y lo con las curvas de aprendizaje mas asimilables. A continuación se enumerarán algunas de las ventajas[27] cada uno de estos lenguajes y se describirá la lógica aplicada a la hora de selección.

Java:

- En Java se ha de declarar una variable y tipificarla antes de poder usarla lo que ayuda a minimizar los errores.
- Este lenguaje es más sencillo de analizar que Python.
- Java es un lenguaje mas portable que Python, ya que se puede usar para programas de escritorio, aplicciones web, e incluso aplicaciones para Android.
- Java es más veloz que Python

Python:

- No es necesario asignar el tipo a una variable cunado se crea, por lo que se pueden usar sin necesidad de especificar de que tipo son y una vez usada se puede reutilizar con un tipeado distinto.

- Python dispone de gran variedad de librerías para diversos tipos de proyecto.
- Este lenguaje es más sencillo de escribir y leer que Java.
- En Python no se utilizan llaves ni puntos y comas si no que se identa lo que hace el lenguaje mucho mas ordenado, sencillo de leer y no se producen errores por falta de llaves o puntos y comas.
- Python es más sencillo de aprender que Java

Para aplicaciones de gran envergadura Java es el lenguaje ideal por su velocidad y potencia, pero para aplicaciones de menor envergadura Python es más legible, mas sencillo de utilizar, dispone de mayor variedad de librerías y mas es fácil de aprender por ello se ha seleccionado Python.

5. DESARROLLO DE LA SOLUCIÓN

En esta sección se detallará el funcionamiento del sistema y su implementación, concretando el hardware construido, el proceso de conexión Bluetooth, y el código fuente diseñado para la aplicación Android y para la Raspberry Pi. Respecto al código fuente se ha adjuntado el código para la Raspberry Pi y trozos relevantes del código de la aplicación Android en los Anexos (ver Anexo A y Anexo B).

5.1. Hardware

Como se detalló en Sección 4.1 el circuito electrónico para la iluminación y calculo de velocidad se lleva a cabo en el proyecto SIITR (I) pero para este proyecto a sido necesario algo de circuitería hasta que se finalice SIITR (I) y se integren ambos sistemas. Se ha emulado lo que seria el sistema de iluminación con un LED azul y se han utilizado un LED rojo y uno verde para los pilotos que indican la velocidad.

5.1.1. GPIO

Los GPIO son pines de entrada y salida de señales digitales, esto quiere decir que un solo pin se puede configurar tanto como entrada para leer de una señal digital, como salida para emitir una. Esto se consigue desde software, al inicio del código se configuran los pines para ser utilizados de un modo u otro, se entrará mas en detalle sobre la configuración de los pines en Sección 5.3.

Como se puede observar en la Figura 5.1 el modelo de Raspberry Pi seleccionado cuenta con 40 GPIOs. De estos 40 pines cuatro son de alimentación -dos proporcionan 5V y dos 3,3V- y ocho pines a tierra. Los 28 pines[28] restantes son los pines programables, de los cuales:

- **PWM generada por hardware:** GPIO12, GPIO13, GPIO18, GPIO19.
- **SPI**
  - SPI0: MOSI (GPIO10); MISO (GPIO9); SCLK (GPIO11); CE0 (GPIO8), CE1 (GPIO7)
  - SPI1: MOSI (GPIO20); MISO (GPIO19); SCLK (GPIO21); CE0 (GPIO18); CE1 (GPIO17); CE2 (GPIO16)
- **I2C:**
  - Data: (GPIO2); Clock (GPIO3)
  - EEPROM Data: (GPIO0); EEPROM Clock (GPIO1)

- **Serial:** TX (GPIO14); RX (GPIO15)

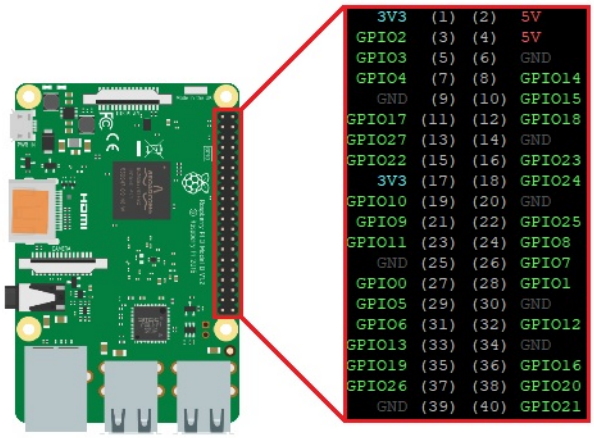


Fig. 5.1. Pines Raspberry Pi 3 Model b+

Para los LEDs verde, rojo y azul se utilizan los pines GPIO5, GPIO6 y GPIO3 respectivamente. Se decide reservar los pines para la conexión Serial (GPIO14 y GPIO15) y los pines de PWM (GPIO12, GPIO13, GPIO18 y GPIO19) por posibles ampliaciones del proyecto que requieran de este tipo de pin.

5.1.2. Especificaciones Eléctricas

La información oficial que los fabricantes de Rasperry Pi proporcionan[29] sobre tensiones y corrientes emitidas y tolerables por los GPIOs es de entre 0V y 3.3V y hasta 16mA, una corriente muy reducida pero suficiente para iluminar un LED. En la Figura 5.2 se pueden observar las especificaciones eléctricas que proporcionan los fabricantes.

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{IL}	Input low voltage[a]	VDD_IO = 1.8V	-	-	0.6	V
		VDD_IO = 2.7V	-	-	0.8	V
		VDD_IO = 3.3V	-	-	0.9	V
V_{IH}	Input high voltage[a]	VDD_IO = 1.8V	1.0	-	-	V
		VDD_IO = 2.7V	1.3	-	-	V
		VDD_IO = 3.3V	1.6	-	-	V
I_{IL}	Input leakage current	TA = +85°C	-	-	5	μA
C_{IN}	Input capacitance	-	-	5	-	pF
V_{OL}	Output low voltage[b]	VDD_IO = 1.8V, IOL = -2mA	-	-	0.2	V
		VDD_IO = 2.7V, IOL = -2mA	-	-	0.15	V
		VDD_IO = 3.3V, IOL = -2mA	-	-	0.14	V
V_{OH}	Output high voltage[b]	VDD_IO = 1.8V, IOH = 2mA	1.6	-	-	V
		VDD_IO = 2.7V, IOH = 2mA	2.5	-	-	V
		VDD_IO = 3.3V, IOH = 2mA	3.0	-	-	V
I_{OL}	Output low current[c]	VDD_IO = 1.8V, VO = 0.4V	12	-	-	mA
		VDD_IO = 2.7V, VO = 0.4V	17	-	-	mA
		VDD_IO = 3.3V, VO = 0.4V	18	-	-	mA
I_{OH}	Output high current[c]	VDD_IO = 1.8V, VO = 1.4V	10	-	-	mA
		VDD_IO = 2.7V, VO = 2.3V	16	-	-	mA
		VDD_IO = 3.3V, VO = 2.3V	17	-	-	mA
R_{PU}	Pullup resistor	-	50	-	65	kΩ
R_{PD}	Pulldown resistor	-	50	-	65	kΩ

[a] Hysteresis enabled
[b] Default drive strength (8mA)
[c] Maximum drive strength (16mA)

Fig. 5.2. Especificaciones Eléctricas Raspberry Pi[29]

Estos datos son incompletos por lo que causan diversos problemas a los desarrolladores a la hora de realizar un diseño, por ello la gente de Mosaic[9] ha elaborado su propia documentación a base de la información proporcionada por los fabricantes de Raspberry Pi y de los procesadores que montan. Ninguno de estos datos es oficial si no que ha sido recopilado para desarrolladores.

[9] «Mosaic Documentation Web» Application notes and technical design information for embedded systems engineers and designers of real time controllers for scientific instruments, industrial control, and process automation. `http://www.mosaic-industries.com/` (Acceso: 25-05-2019)

GPIO input/output pin electrical characteristics	
Output low voltage V_{OL}	<0.40
	<0.66
	<0.40
	<0.40
Output high voltage V_{OH}	>2.40
	>2.64
	>2.90
Input low voltage V_{IL}	<0.80
	<0.54
	<1.15
Input high voltage V_{IH}	>2.00
	>2.31
	>2.15

TABLA 5.1. ESPECULACIÓN SOBRE ESPECIFICACIONES ELÉCTRICAS DE GPIO[30]

Es necesario tener en cuenta toda la información sobre especificaciones eléctricas para no dañar la Raspberry Pi. A pesar de que el presente circuito es relativamente sencillo y se nutre de la propia alimentación de la Raspberry, la circuitería final que se desarrollará en SIITR (I) dispondrá de su propia alimentación y las señales que se envíen a la Raspberry Pi deberán respetar estas limitaciones para no dañar la placa.

5.1.3. Circuito y componentes

Para el circuito el diseño consiste de un LED y una resistencia. Un LED es un dispositivo semiconductor capaz de emitir luz ante una diferencia de potencial suficiente entre sus extremos. En función del color del LED este requerirá mayor o menor tensión para iluminarse, para la iluminación se han utilizado un LED rojo, uno verde y uno azul con las siguientes características:

Color LED	Log. de onda	Voltaje (V)	Corriente (A)
Rojo	610< λ<760	2	
Verde	500< λ<570	2.3	15
Azul	450< λ<500	2.8	

TABLA 5.2. ESPECIFICACIONES LEDS

Como cada LED requiere un voltaje diferente se ha de calcular un resistencia concreta para cada uno. Teniendo en cuenta los datos proporcionados por el fabricante de

Raspberry Pi detallados en Sección 5.1.2 un pin proporciona 3.3V por lo que los cálculos quedan del siguiente modo.

$$R_{Rojo} = \frac{V_{Pin} - V_{LED}}{I_{LED}} = \frac{3{,}3 - 2}{0{,}015} = 86{,}6\Omega \tag{5.1}$$

$$R_{Verde} = \frac{V_{Pin} - V_{LED}}{I_{LED}} = \frac{3{,}3 - 2{,}3}{0{,}015} = 66{,}6\Omega \tag{5.2}$$

$$R_{Azul} = \frac{V_{Pin} - V_{LED}}{I_{LED}} = \frac{3{,}3 - 2{,}8}{0{,}015} = 33{,}3\Omega \tag{5.3}$$

Si se quisieran adquirir exactamente esas resistencias tendrían un precio muy elevado por lo que se seleccionarán resistencias con valores normalizados con una tolerancia del 5 %. Cuanto menor la resistencia mayor la corriente por el LED lo que permite mayor luminosidad por ello se ha optado por seleccionar la resistencia inmediatamente menor a la calculada.

x 1	x 10	x 100	x 1.000 (K)	x 10.000 (10K)	x 100.000 (100K)	x 1.000.000 (M)
1 Ω	10 Ω	100 Ω	1 KΩ	10 KΩ	100 KΩ	1 M Ω
1,2 Ω	12 Ω	120 Ω	1K2 Ω	12 KΩ	120 KΩ	1M2 Ω
1,5 Ω	15 Ω	150 Ω	1K5 Ω	15 KΩ	150 KΩ	1M5 Ω
1,8 Ω	18 Ω	180 Ω	1K8 Ω	18 KΩ	180 KΩ	1M8 Ω
2,2 Ω	22 Ω	220 Ω	2K2 Ω	22 KΩ	220 KΩ	2M2 Ω
2,7 Ω	27 Ω	270 Ω	2K7 Ω	27 KΩ	270 KΩ	2M7 Ω
3,3 Ω	33 Ω	330 Ω	3K3 Ω	33 KΩ	330 KΩ	3M3 Ω
3,9 Ω	39 Ω	390 Ω	3K9 Ω	39 KΩ	390 KΩ	3M9 Ω
4,7 Ω	47 Ω	470 Ω	4K7 Ω	47 KΩ	470 KΩ	4M7 Ω
5,1 Ω	51 Ω	510 Ω	5K1 Ω	51 KΩ	510 KΩ	5M1 Ω
5,6 Ω	56 Ω	560 Ω	5K6 Ω	56 KΩ	560 KΩ	5M6 Ω
6,8 Ω	68 Ω	680 Ω	6K8 Ω	68 KΩ	680 KΩ	6M8 Ω
8,2 Ω	82 Ω	820 Ω	8K2 Ω	82 KΩ	820 KΩ	8M2 Ω
						10M Ω

Fig. 5.3. Tabla de resistencias normalizadas[10]

Con estos datos se puede proceder al montaje del circuito.

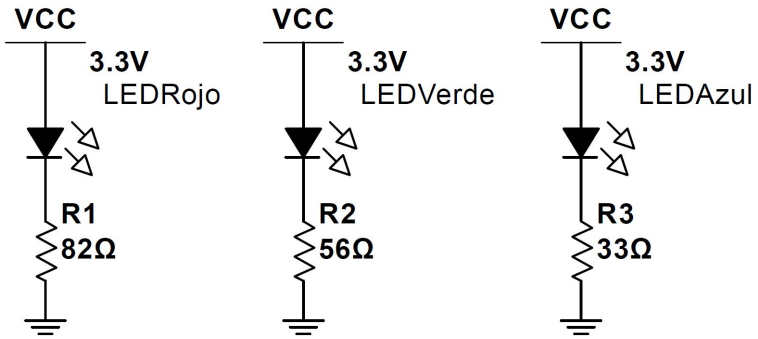


Fig. 5.4. Circuito LEDs

[10]«Valores comerciales de resistencias» *Electrontools* `https://www.electrontools.com/Home/WP/2016/04/14/valores-comerciales-de-resistencias/` (Acceso: 25-05-2019)

5.1.4. Montaje final

Habiendo dimensionado los componentes necesarios en Sección 5.1.3 y habiendo seleccionado los pines donde conectarlos en Sección 5.1.1 el ultimo paso es el montaje. Se debe conectar el circuito para el LED verde en el pin GPIO5, el del LED rojo en el GPIO6 y por ultimo el circuito del LED azul en el pin GPIO3 (ver Figura 5.5).

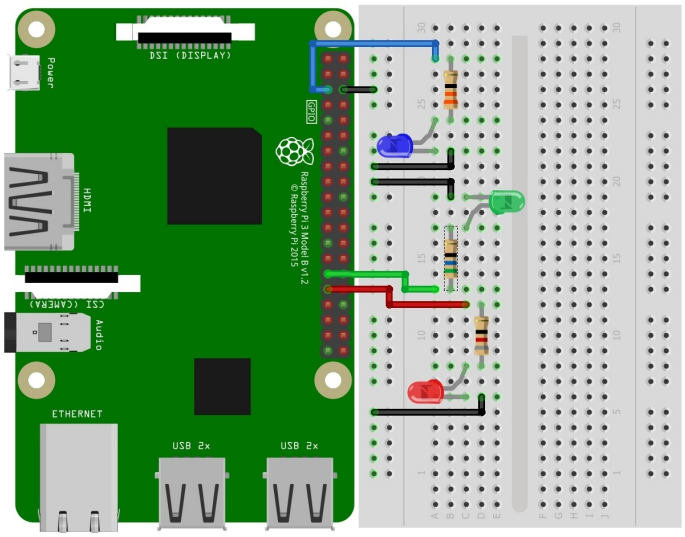


Fig. 5.5. Conexión Circuito-Raspberry Pi

Es importante que los colores de los LEDs se correspondan con sus respectivos pines. Esto se debe a que para su encendido y apagado no se distingue por color sino por pin, por ejemplo en caso de confundir los pines del LED rojo y verde se causaría que el sistema funcionara de manera contraria y el LED verde se encendería cuando se superara el limite y el rojo cuando se circulara a una velocidad adecuada. Si por otro lado se conectaran los LEDs en pines que no están asignados a ninguna salida o entrada utilizada en este proyecto este LED no se iluminaría.

5.2. Conexión Bluetooth

En una conexión Bluetooth existen dos roles, maestro y esclavo[21]. En toda conexión Bluetooth uno de los dispositivos debe hacer la función de esclavo y uno la de maestro. El maestro se encarga de enviar la información del reloj para sincronizarse y la de frecuencia de los saltos para poder realizar la conexión y el esclavo se dedica a sincronizarse y seguir la frecuencia de saltos establecida por el maestro. En el presente proyecto la función de esclavo la tiene la Raspberry Pi mientras la de maestro recae sobre el dispositivo móvil.

Al establecerse una conexión[31] con un dispositivo por primera vez estos se sincronizan y la información sobre su clase, dirección MAC y el nombre se guardan para poder establecer conexión una próxima vez sin tener que pasar por una fase de búsqueda de

dispositivos cercanos. Una vez sincronizados los dispositivos conocen su mutua existencia, comparten una clave de vinculación y son capaces de establecer conexiones cifradas entre ellos. Actualmente no se ha desarrollado el software necesario para la búsqueda de dispositivos y su sincronización por lo que las aplicaciones desarrolladas requieren que Raspberry y dispositivo móvil ya se encuentren sincronizados para poder establecer la conexión.

Una vez sincronizados los dispositivos la conexión entre ellos se realiza mediante un canal de transmisiones compartido, los *sockets* son el mecanismo que permite el establecer conexión entre dos aplicaciones independientes que se ejecutan en maquinas distintas, en este caso el script en Python que se ejecuta en la Raspberry Pi y la aplicación Android que se ejecuta en el dispositivo móvil. El *socket* creado es de tipo RFCOMM, que es el protocolo de conexión dedicado a emular un puerto serie RS-232. Al ser un protocolo orientado a la conexión se requiere que una aplicación haga la función de *server* y la otra la de *client*.

En la conexión ambas aplicaciones deben crear un *socket*[32] de conexión, el *server* es la aplicación pasiva, se dedica a «escuchar» a la espera de recibir una solicitud de conexión, mientras el *client* es que que solicita la conexión. Para este proyecto el *server* es la Raspberry Pi y el *client* es el dispositivo móvil.

El *server* crea un *socket* para la conexión y asocia dicho *socket* con un puerto y/o un host (*bind*), una vez hecho esto comienza a «escuchar» (*listen*). Por ultimo se bloquea la aplicación hasta recibir una solicitud de a conexión (*accept*). El *client* crea un *socket* y establece conexión con un puerto o dirección especificado (*connect*).

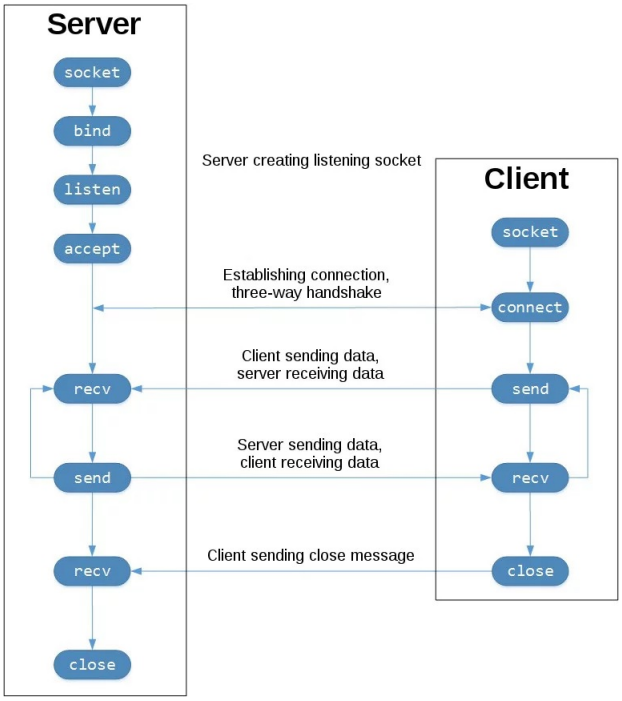


Fig. 5.6. Diagrama de flujo de conexión Bluetooth[32]

En resumen, el proceso de conexión requiere que los dispositivos estén previamente emparejados de esta manera la Raspberry Pi, que se encuentra a la espera de una conexión entrante (*server*) aparecerá en la lista de dispositivos en la aplicación Android. El usuario entonces procede a seleccionarla de la lista enviando la dirección MAC de la Raspberry a la actividad principal (pantalla del mapa ver Figura 5.11) donde, la aplicación (*client*) establecerá la conexión Bluetooth.

5.2.1. Server: Raspberry Pi

La Raspberry Pi es el *server* de la conexión, se creará un *socket*, de tipo RFCOMM y se quedará a la espera de una conexión «escuchando» todos los puertos disponibles. Una vez establecida una conexión imprimirá la dirección MAC del dispositivo con el que se ha conectado y se iniciará el flujo de la aplicación. También se ha implementado un control de errores en caso de una conexión fallida, en ese caso se cerrará el *socket* y el *client* de la conexión fallida y se creará un *socket* nuevo para volver a intentar una conexión (ver Código 2).

5.2.2. Client: Dispositivo Móvil

La aplicación en Android ejecuta el proceso de *client*, para establecer conexión el primer paso es obtener la lista de dispositivos emparejados con el teléfono móvil. Para obtener la lista de dispositivos se debe crear un objeto BluetoothAdapter, que mediante el método getBondedDevices se obtendrá toda la información sobre los dispositivos emparejados y se imprimirán por pantalla para que el usuario pueda seleccionar a que dispositivo conectarse (ver Código 3).

En la Figura 5.7 se puede observar la lista de dispositivos enlazados en la configuración de Android y la lista de dispositivos con sus direcciones MAC en la aplicación desarrollada.

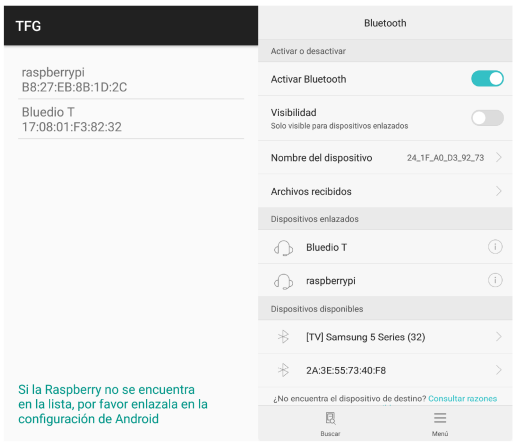


Fig. 5.7. Lista de dispositivos emparejados

En la aplicación el usuario puede seleccionar el dispositivo al que conectarse, cuando

el usuario pulsa un dispositivo de la lista se activa el método implementado en Código 4. Este método intenta iniciar la actividad principal de la aplicación (la pantalla con el mapa, ver Figura 5.11) y envía el dato de la dirección MAC del dispositivo seleccionado por el usuario para que dicha actividad intente establecer la conexión Bluetooth. Este método también pone el texto «Conectando...» para que el usuario sepa que se iniciará el intento de conexión (ver Código 4).

Una vez que la actividad a enviado la dirección MAC del dispositivo seleccionado se inicia el proceso de conexión. Para ello la actividad principal recupera la dirección MAC (que se ha pasado como parámetro en el cambio de actividad), y se crea un objeto device donde se almacenará la información del dispositivo seleccionado por el usuario. Al igual que se hizo en el *server*, el *client* debe crear un *socket* (ver Código 5). A diferencia del *server* que crea un canal y se queda a la «escucha» de posibles conexiones, el *socket* del *client* apunta directamente al dispositivo seleccionado por el usuario. Como se explicó en Sección 5.2 el *client* es el que lleva a cabo la acción de conectarse por lo que ahora se inicia el proceso de conexión. Se ha añadido un envío de un dato para comproba que la conexión es correcta, de no serlo el envío del dato fallaría y se cerraría la conexión volviendo a la actividad de la lista de dispositivos emparejados (ver Código 6).

5.2.3. Codificación utilizada

Para enviar la información necesaria vía Bluetooth se ha decidido codificarla, esto simplifica el flujo de información ya que en vez de enviarse una palabra o una frase se envía solo un caracter. Esto además añade un factor de seguridad, ya que a pesar de que una conexión Bluetooth la información viaja cifrada codificándola se evita que se entienda la información enviada a primera vista. En Tabla 5.3 es especifica cual es el flujo de información, que código se envía y su significado. Actualmente el envió de información es unidireccional -desde la aplicación Android hasta la Raspberry Pi- pero en un futuro (ver Sección 4.1) esto no será así por lo que se podrá usar un mismo código para tareas diferentes.

Dirección	Código	Significado
Aplicación Android - Raspberry Pi	0	Encender el Sistema de Iluminación
Aplicación Android - Raspberry Pi	1	Apagar el Sistema de Iluminación
Aplicación Android - Raspberry Pi	f	El usuario no supera el limite de velocidad
Aplicación Android - Raspberry Pi	t	El usuario supera el limite de velocidad
Aplicación Android - Raspberry Pi	q	Terminar conexión Bluetooth

TABLA 5.3. CODIFICACIÓN INFORMACIÓN ENVIADA POR BLUETOOTH

5.3. Software para Raspberry Pi

En esta sección se detallará el funcionamiento del código fuente desarrollado para la Raspberry Pi a la vez que se entrará en detalle en aspectos importantes sobre su desarrollo.

5.3.1. Requisitos

Para el correcto funcionamiento del código se requiere que el dispositivo móvil y la Raspberry Pi estén previamente emparejados. En cuanto a requerimientos de hardware no es necesario nada mas que lo enumerado en Sección 4.2.2 y respecto al software se necesita que la Raspberry Pi tenga instalado Python.

5.3.2. Funcionamiento

La Raspberry se encarga de la conexión entre el Sistema de Iluminación y el de Navegación mediante Bluetooth. Al iniciarse el proceso (ver Figura 5.8) el primer paso es establecer la conexión Bluetooth con el dispositivo móvil, no se continua con el flujo del la aplicación hasta que se completa la conexión. Una vez completada la aplicación Android puede enviar información.

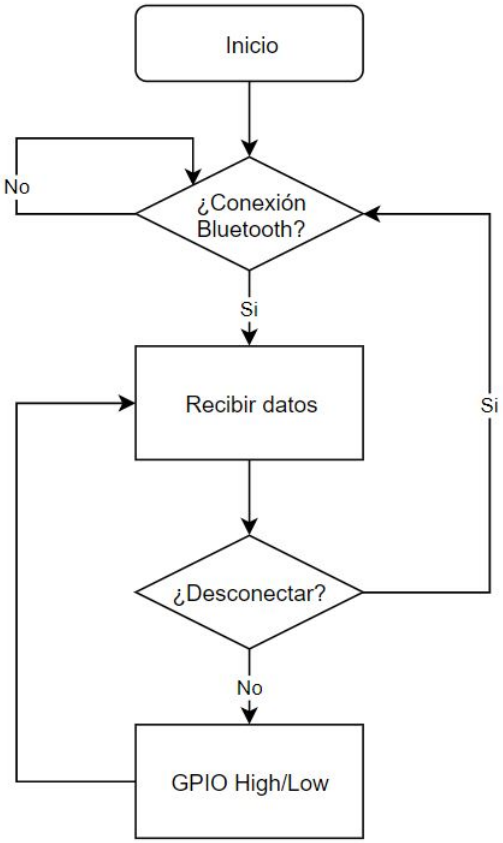


Fig. 5.8. Diagrama de flujo del código para Raspberry Pi

Para programar la Raspberry se utiliza Python, la aplicación se fundamentará en dos pilares, la conexión Bluetooth (ver Sección 5.2) y el control de los LEDs, continuación se describirá el funcionamiento de esta segunda parte.

En Tabla 5.3 se especifica el significado de cada código que se puede recibir por Bluetooth. Si se recibe un «0» se apaga el Sistema de Iluminación -para este caso el Sistema de Iluminación viene representado por un LED azul-, si se recibe un «1» se enciende. Una «f» significa que el usuario ha dejado de superar el limite por lo que se apaga el LED rojo y se enciende el verde y una «t» lo contrario, que el usuario ha empezado a superar el limite y se debe apagar el LED verde y encender el rojo. Por ultimo una «q» implica que el usuario desea finalizar la conexión Bluetooth con la Raspberry por lo que se apagan todos los LEDs y se cierra la conexión (ver Código 1).

5.4. Software para aplicación Android

En esta sección se detallará el funcionamiento del código fuente desarrollado para la aplicación Android a la vez que se entrará en detalle en los aspectos más importantes sobre su desarrollo.

5.4.1. Requisitos

Para el correcto funcionamiento del código se requiere que el dispositivo móvil y la Raspberry Pi estén previamente emparejados. En cuanto a requerimientos de hardware no es necesario nada mas que lo enumerado en Sección 4.2.1 y respecto al SO se requiere que el dispostivo que vaya a ejecutar la aplicación disponga de una versión de Android igual o superior a la 4.0.

5.4.2. Funcionamiento

Al igual que la Raspberry la aplicación Android también se dividirá en dos pilares, el de la conexión Bluetooth (ver Sección 5.2) y el trazado de ruta y obtención de la velocidad.

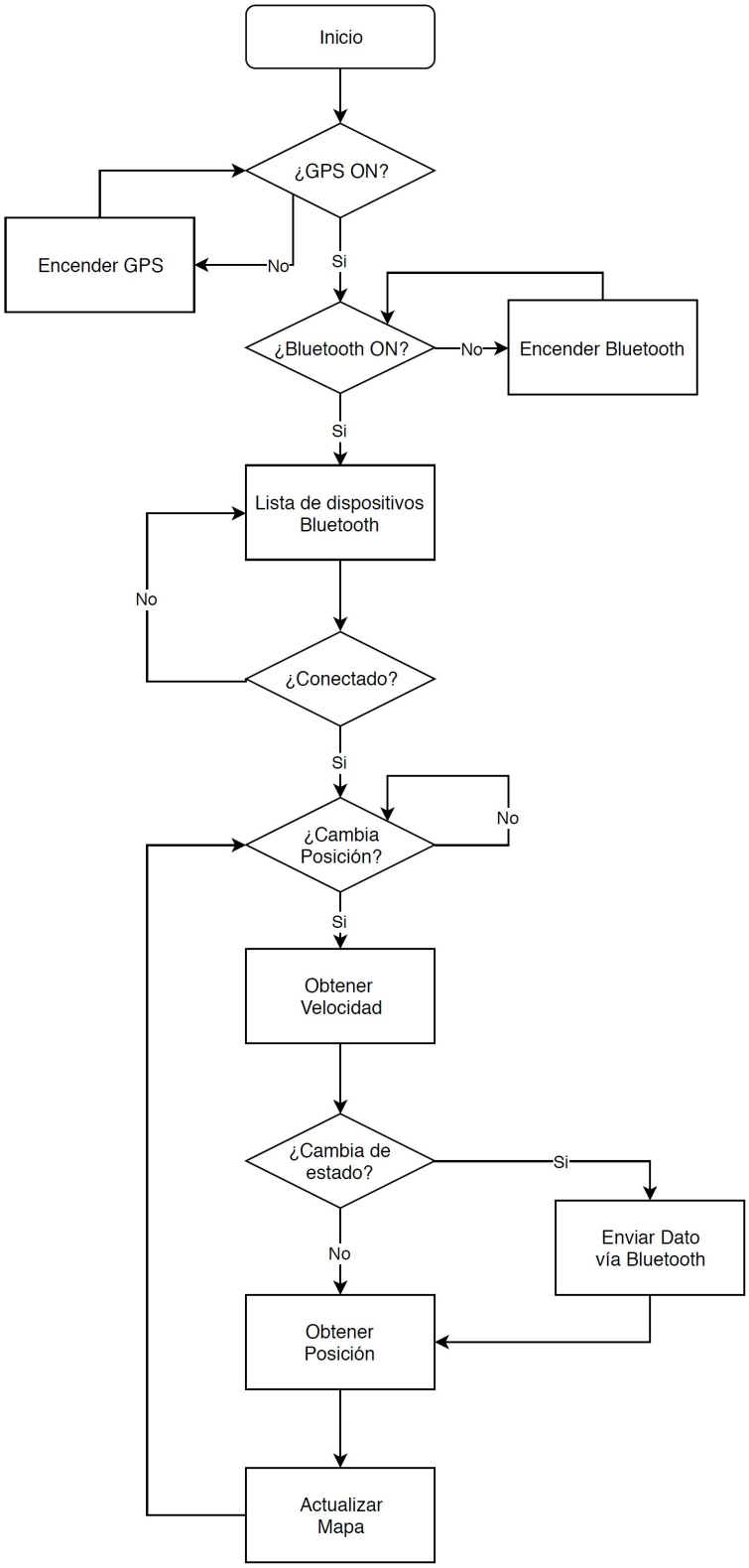


Fig. 5.9. Diagrama de flujo de la Aplicación Android

La aplicación se inicia comprobando el estado del Bluetooth y del GPS, estos dos parámetros deben estar activados en la configuración de Android para el correcto funcio-

namiento de la aplicación. La primera vez que se ejecute la aplicación el usuario debe proporcionarle permisos a esta para el acceder a la ubicación. Una vez garantizado este permiso no se volverá a preguntar al usuario a menos que este prohíba específicamente a la aplicación a acceder a la ubicación.

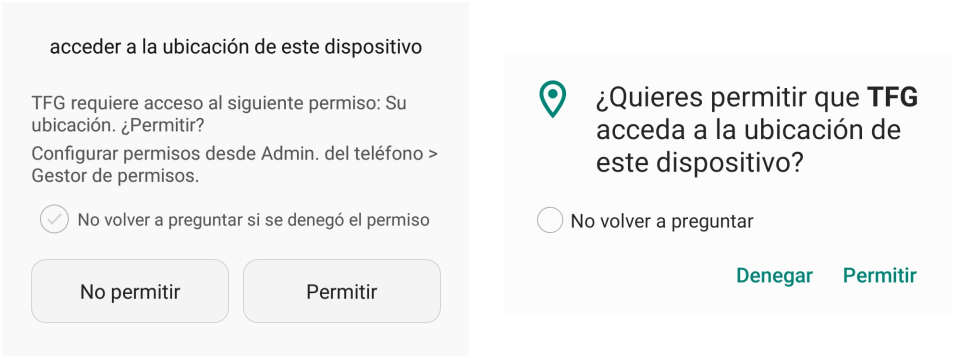


Fig. 5.10. Permisos de acceso a la ubicación del dispositivo en dos versiones de Android diferentes

Una vez que Bluetooth y GPS están activados se muestra la primera pantalla de la aplicación con la lista de dispositivos, cuando el usuario selecciona un dispositivo se establece conexión con este y se abre la pantalla con el mapa y el dato de velocidad.

Fig. 5.11. Conexión Bluetooth

La actividad principal es la encargada de la obtención de la velocidad y la ubicación. Al obtener la ubicación se traza la ruta y se calcula la velocidad, esta se imprime por pantalla para que el usuario pueda verla. Los botones «LED ON» y «LED OFF» son los que representan el encendido y apagado del sistema de iluminación. Si el usuario pulsa alguno de los botones se envía un dato vía Bluetooth y como se detallo en Sección 5.2 la Raspberry recibe y lo procesa.

En caso de que el usuario pulse el botón de «DESCONECTAR BLUETOOTH» se finaliza la conexión, se cierra la pantalla del mapa y se vuelve a la pantalla con la lista de dispositivos.

Comprobación de Bluetooth y GPS

Antes de comprobar si el Bluetooth y el GPS se encuentran activos se hace una verificación de que el dispositivo tiene un chip Bluetooth, esto se comprueba creando un objeto de tipo BluetoothAdapter. Este objeto es el que se utilizó para obtener la lista de dispositivos e iniciar la conexión Bluetooth (ver Sección 5.2.2).

Una vez comprobado que el dispositivo tiene Bluetooth se comprobará si esta activado. Para ello se utiliza el objeto tipo BluetoothAdapter creado anteriormente, que mediante el método isEnabled se conseguirá la información sobre el estado del Bluetooth. Si esta activado se puede continuar con el flujo de la aplicación sin problema pero si esta desactivado se ha de solicitar al usuario que lo active, para ello se utilizará este mismo objeto y se instanciará un método gestionado por el SO para dirigir al usuario a los ajustes (ver Código 8).

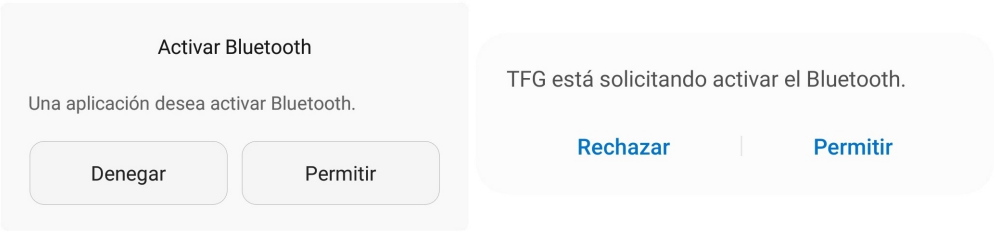


Fig. 5.12. Solicitud de activación de Bluetooth en dos dispositivos diferentes

Al utilizarse el propio SO para esta solicitud el cuadro que se muestra en Figura 5.12 variará en función de la versión de Android y de las diferentes capas de personalización que utilice el dispositivo sobre el que se ejecute la aplicación. En la Figura 5.12 se muestra el cuadro de solicitud de activacon del Bletooth para Android 6.0 y Android 9.0.

Para comprobar el estado del GPS se seguirá la misma linea de actuación que con el Bluetooth con diferencias a la hora de la solicitud de activación. Para la comprobación del estado del GPS se utiliza un objeto de tipo LocationManager y el método isProviderEnabled (ver Código 8).

En caso de que esté activado se continua el flujo de la aplicación, en caso contrario se mostrará un cuadro de dialogo solicitándole al usuario que active el GPS. El cuadro de dialogo (AlertDialog) diseñado no varia en función de la versión de Android pero si puede llegar a sufrir alguna modificación si el dispositivo utiliza alguna capa de personalización. Este cuadro de dialogo muestra un mensaje informando al usuario que el GPS esta desconectado y un botón que le dirige a los ajustes de Android para activarlo.

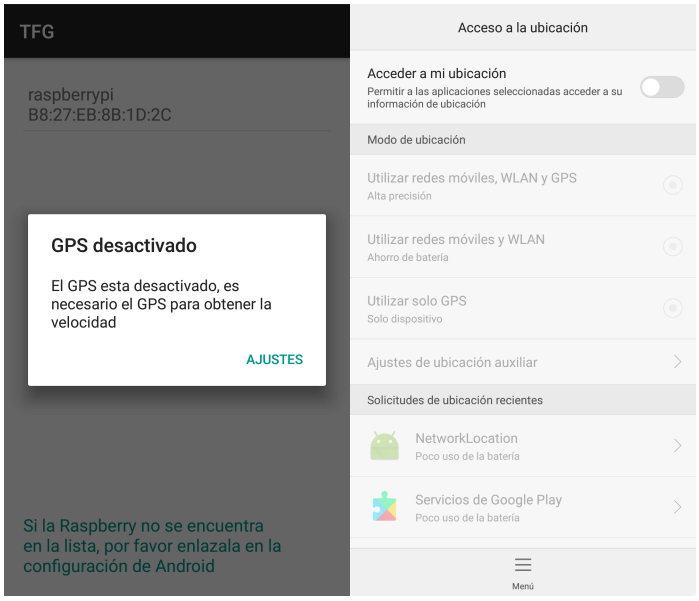


Fig. 5.13. Solicitud de activación del GPS

Obtención de posición y velocidad y trazado de ruta

Para la obtención de la posición y velocidad se necesita un objeto de tipo LocationManager, este objeto podrá utilizar los métodos necesarios par la obtención de estos datos. La creación de este objeto permite utilizar un interrupción configurable que al activarse llama al método onLocationChanged. Esta interrupción es la que se utilizará para actualizar mapa y velocidad, para utilizarla se pueden configurar los siguientes parámetros:

- Tiempo: Este parámetro permite que la interrupción se active cada vez que pase el tiempo concretado.
- Distancia: Este parámetro permite que la interupcion se active cada vez que la diferencia de posición entre el punto actual y el anterior sea la distancia especificada.

Para el desarrollo se ha decidido no concretar una distancia mínima sino por obtener la posición cada intervalo de tiempo. Se ha optado por este método ya que la velocidad del usuario que utilice el sistema rondará los 10Km/h por lo que el rango de distancias para activar a interrupción debería ser inferior a un metro y el GPS no es capaz de proporcionar esta precisión. La selección de un intervalo de tiempo para la obtención de la posición permite que cuando el desplazamiento es mínimo, se consiga trazar una ruta (ver Código 9).

Ambos métodos producen resultados erráticos cuando el usuario tiene velocidad nula o muy baja, pero esto es debido a la precisión capaz de proporcionar el GPS. La precisión es mejorable según el dispositivo que se utilice, si se utiliza uno compatible con GPS, GLONASS y Galileo la precisión será mejor que utilizando solo GPS o solo GPS y GLONASS (ver Sección 6.3).

Cada vez que pasa el tiempo establecido en la configuración anteriormente mencionada se activa el método onLocationChanged que trae un objeto tipo location. Este objeto incluye como atributos la longitud y la latitud lo que representa un punto sobre la superficie de la tierra. Este objeto tiene un método que calcula la velocidad mediante la distancia entre el punto actual y el anterior y el tiempo que ha pasado entre ellos de esta manera se obtiene la velocidad del usuario.

Se ha creado una variable llamada speedState que controla el estado de la velocidad del usuario. Si el usuario supera el limite de velocidad speedState esta en TRUE, si por el contrario el usuario tiene una velocidad adecuada speedState es FALSE. Al obtener el dato de velocidad esta variable se tiene en cuenta, ya que si speedState es TRUE y la velocidad que se obtiene supera el limite no es necesario mandar el comando de activación del LED rojo porque ya esta activado. En cambio si el dato de velocidad que se obtiene no supera el limite sí se enviará el dato vía Bluetooth ya que se trata de un cambio de estado. Esta variable lo que permite es reducir el envió de información vía Bluetooth limitándola a un cambio de estado, de estar superando el limite a dejar de superarlo y viceversa (ver Código 10).

Para el trazado de la ruta se utiliza el mismo objeto location, el método empleado para el trazado de ruta es updateTrack que une el punto que recibe con el ultimo punto dibujado sobre el mapa. Los puntos que se van recibiendo se guardan en una lista de datos, esta lista actualmente solo hace la función de registrar la ruta pero en uno de los objetivos futuros del proyecto se pretende guardar las rutas del usuario para permitirle visualizarlas en cualquier momento y compartirlas con otros usuarios de la aplicación (ver Código 11).

Para poder dibujar el mapa en la aplicación se ha utilizado los mapas proporcionados por la API de Google Maps. Una API es un medio que permite integrar una serie de servicios en los desarrollos. En este caso lo que se necesita es dibujar un mapa según la ubicación en la que se encuentre el usuario y dibujar una linea sobre el mismo. Para poder utilizar la API se debe disponer de una clave que identifica al desarrollador y a la aplicación a la hora de hacer solicitudes a la API. Para obtener la clave el desarrolador debe registrarse en Google Cloud Platform y seleccionar la API que desea utilizar. Una vez seleccionada la API se debe vincular la aplicación con su cuenta y obtendrá una clave que solo se puede utilizar desde la aplicación concretada.

La API es gratuita si los picos de uso son reducidos, aun así Google proporciona un crédito mensual de 200 USD[11] tras los cuales se ha de empezar a pagar por uso. Como el proyecto esta en fase de desarrollo los picos de uso no llegan al minino facturable pero de extenderse el uso de la aplicación habría que comenzar a pagar por su uso.

[11] Google Cloud, `https://cloud.google.com/maps-platform/pricing/` (Acceso: 01-06-2019)

Diseño Responsive

El Diseño Responsive es una filosofía de diseño con el objetivo de que todas las interfaces diseñadas se puedan apatar a cualquier tipo y tamaño de pantalla. Debido a que Android no fue diseñado para un tipo concreto de dispositivo existen multitud de tamaños, formatos y resoluciones de pantalla de los dispositivos que utilizan Android como SO por lo que la aplicación ha sido diseñada teniendo esto en cuenta.

Para la pantalla del listado se ha colocado el texto informativo para que quede en la parte inferior de la pantalla y sobre este se coloca el texto que aparece cuando se inicia el proceso de conexión. La lista de dispositivos quedará en la parte superior ocupando el resto de la pantalla. Todos los objetos en la pantalla se colocarán centrados ocupando la totalidad de su anchura dejando un margen de 20pt tanto a los lados como en las partes superior e inferior (ver Código 12).

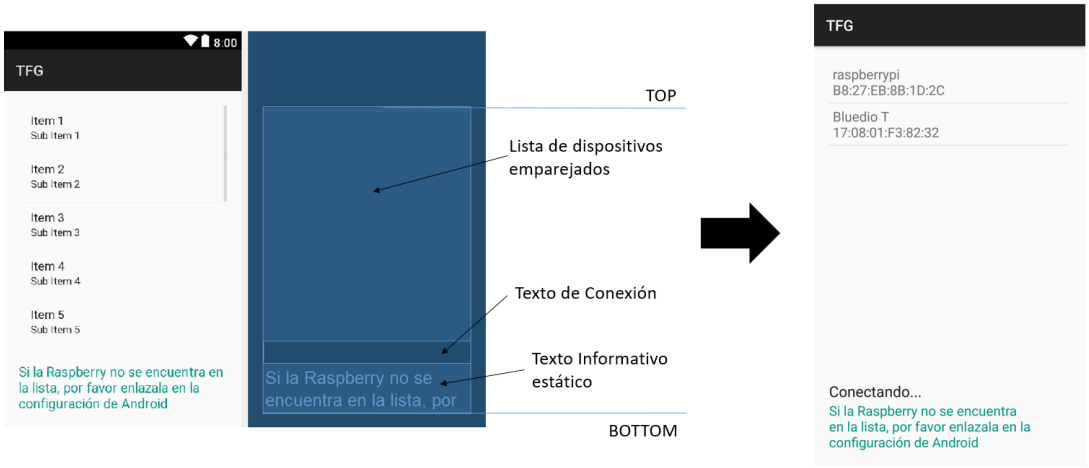


Fig. 5.14. Pantalla 1: Lista de dispositivos

Para la pantalla del mapa se han colocado dos botones en la parte superior de la pantalla, bajo ellos dos bloques de texto uno debajo del otro. En la parte mas baja de la pantalla se ha colocado el botón de desconectar Bluetooth y entre el bloque de texto mas bajo y el botón de desconectar Bluetooth se ha colocado el mapa para que rellene todo el hueco restante. Al igual que en la anterior pantalla todos los objetos dejan un margen de 20pt por todos los bordes de la pantalla y entre si mantienen un margen de 2pt (ver Código 13).

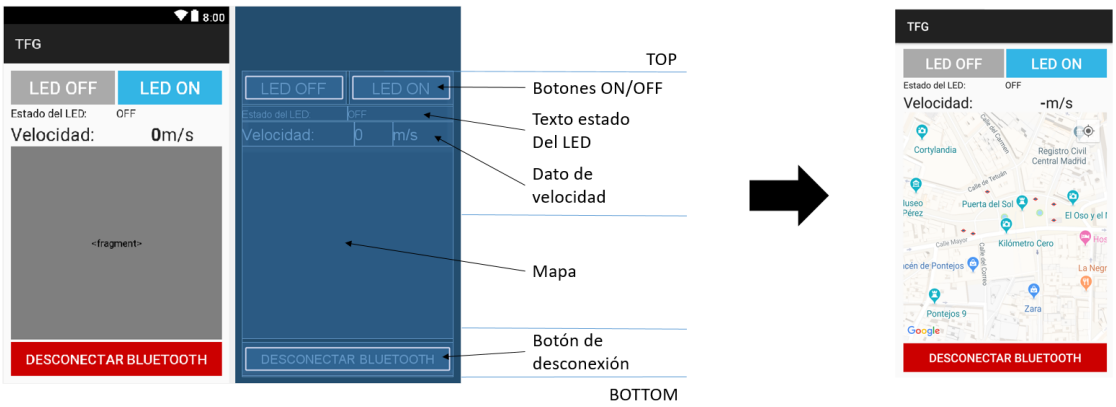


Fig. 5.15. Pantalla 2: Mapa

5.5. Utilización de una Raspberry para la lectura de la velocidad

Para el envío del dato de velocidad desde el SIITR (I) al SIITR (II) se ha decidido utilizar una señal modulada por ancho de pulsos (PWM), este tipo de señal permite enviar información mediante una señal cuadrada que la Raspberry puede interpretar. El objetivo de esta sección es evaluar si la Raspberry Pi es capaz de leer el ancho de pulso de la señal PWM y si el error de lectura es un error asumible.

5.5.1. Funcionamiento

El funcionamiento diseñado consiste en que el SIITR (I) envíe una señal PWM con el dato de la velocidad del usuario y la Raspberry midiendo los anchos de pulso de la señal sea capaz de obtener la velocidad, procesar la información y encender o apagar los LEDs rojo y verde y enviar el dato de velocidad vía Bluetooth para que la aplicación Android lo imprima por pantalla.

El SIITR (I) será capaz de medir la frecuencia de giro de la rueda del vehículo que esté usando el usuario, y conociendo el radio de la rueda podrá calcular la velocidad del usuario mediante la formula en la Ecuación 5.4. En esta formula la v representa la velocidad lineal del usuario, R el radio de la rueda y f la frecuencia de giro de la rueda.

$$v = \frac{2\pi R}{T} = 2\pi R f \tag{5.4}$$

Una vez calculada la velocidad el SIITR (I) generará una señal PWM como se muestra en la Figura 5.16, donde la señal en rojo en la gráfica superior representa la velocidad del usuario. La frecuencia de la señal será constante y el ancho de pulso variará en función de la velocidad, a mayor velocidad mas ancho es el pulso y viceversa. Una mayor frecuencia en la PWM significa mayor velocidad de actualización del dato pero menor precisión, mientras que una frecuencia menor permite un dato de velocidad mas preciso pero mas lento en actualizar.

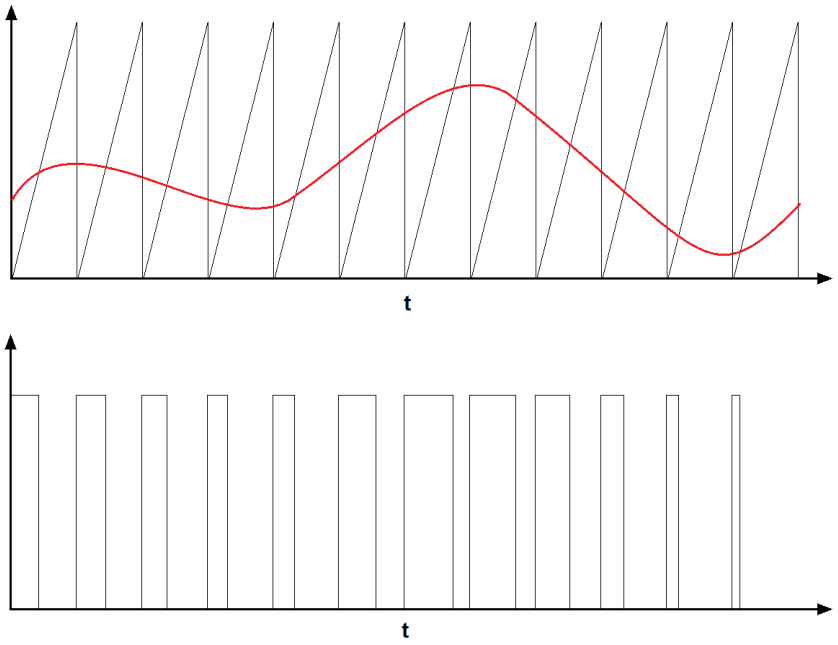


Fig. 5.16. Funcionamiento de una señal PWM

El ciclo de trabajo de la PWM es la diferencia entre el tiempo en zona alta y el tiempo en zona baja (ver Figura 5.17) cuando la velocidad es máxima del ciclo de trabajo es del 100 % y cuando la velocidad es nula es un 0 %.

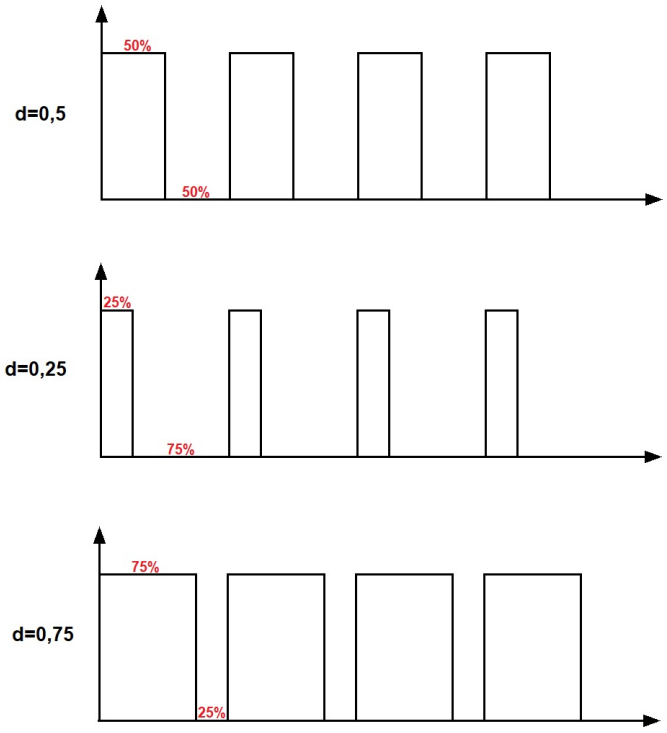


Fig. 5.17. Ciclo de trabajo de una señal PWM

Para que la Raspberry sepa que velocidad lleva el usuario ha de calcular el ciclo de trabajo de la señal que recibe, conociendo este dato y la velocidad máxima la velocidad actual se calcula como se muestra en Ecuación 5.5.

$$v = v_{max.}d \tag{5.5}$$

5.5.2. Viabilidad de la solución

Aunque en teoría el diseño es viable se debe hacer un estudio para comprobar si la Raspberry Pi es capaz de ejercer la función de lectura y cuales son los errores que se producirán. Como ya se mencionó anteriormente la documentación proporcionada por los fabricantes de Raspberry Pi es incompleta por lo que para evaluar si la Raspberry puede hacer la función de lectura. Se han realizado una serie de pruebas con diferentes frecuencias y anchos de pulso (ver Anexo C).

Frecuencia (Hz)	Ciclo de trabajo	Error medio (ms)	Error máximo (ms)
50	0,5	0,074	0,152
100	0,9	0,075	0,197
100	0,5	0,069	0,147
100	0,3	0,069	0,141
200	0,5	0,070	0,164
200	0,1	0,066	0,193
300	0,1	0,099	0,192
500	0,1	0,061	0,097

TABLA 5.4. RESULTADOS DE MEDIDAS DE SEÑALES PWM

Los resultados obtenidos se detallan en Tabla 5.4. Se puede concluir que los errores medios son similares para todas las frecuencias, es decir, el error no depende de la frecuencia utilizada, el error medio de todas las medidas es 0,073ms. Este error puede ser asumible si la frecuencia de la PWM es adecuada.

Frecuencia (Hz)	Ciclo de trabajo	Tiempo teorico HIGH (ms)	Error
50	0,1	2,00	3,65 %
100	0,1	1,00	7,30 %
200	0,1	0,50	14,60 %
300	0,1	0,33	21,90 %
500	0,1	0,20	36,50 %

TABLA 5.5. ERROR PORCENTUAL SEGUN LA FRECUENCIA DE LA PWM

En Tabla 5.5 se han calculado los errores porcentuales según diversas frecuencias. Cuanto menor sea la velocidad, menor será el ciclo de trabajo mayor el error por lo que para este calculo se ha seleccionado un ciclo de trabajo del 10 %.

El error si se utilizara un frecuencia de 500Hz para la señal PWM seria del 36,5 % sobre la velocidad real, si la velocidad limite de circulación es 10Km/h y se supusiera que a esta velocidad el ciclo de trabajo e la PWM es del 10 % el error para esta velocidad sería de 3,7Km/h, este error es muy elevado por lo tanto no se puede asumir. En cambio para una frecuencia de 50Hz y 10Km/h el error sería de menos de 0,37 Km/h por lo que se podría aceptar.

Para mejorar la precisión ademas de elegir una frecuencia baja se debe elegir un máximo de velocidad medible adecuado, es decir, si se selecciona un máximo de velocidad muy elevado para bajas velocidades (ciclos de trabajo pequeños) los errores de medida serán muy elevados (ver Figura 5.18) , pero si se selecciona un rango, por ejemplo, de 0 Km/h a 20 Km/h para la velocidad de 10Km/h el ciclo de trabajo será del 50 % por lo que el error se reduce al 0,73 % lo que serían 0,073 Km/h.

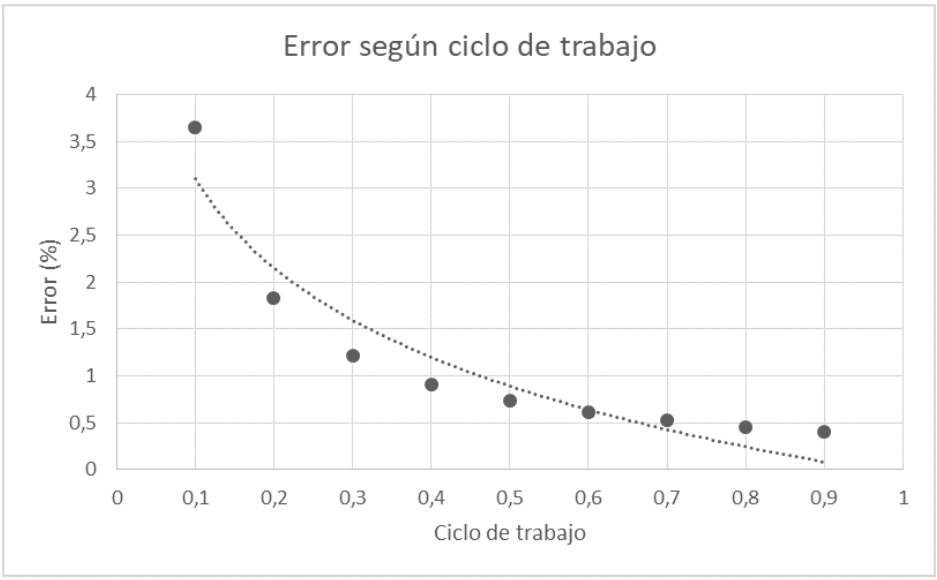


Fig. 5.18. Error de una PWM de 50Hz según el ciclo de trabajo

6. RESULTADOS EXPERIMENTALES

En esta sección se recopilarán una serie de pruebas realizadas en diferentes dispositivos y con distintas velocidades y ubicaciones. Para estas pruebas se han utilizados dos dispositivos móviles diferentes tanto en hardware como en software.

Se han realizado una serie de pruebas para comprobar la adaptabilidad de la aplicación en diferentes pantallas, la velocidad y robustez de la conexión Bluetooth y GPS, la precisión y velocidad de obtención de la posición, la precisión de la velocidad obtenida y por ultimo la precisión del trazado de la ruta recorrida.

Principalmente se han utilizado dos dispositivos para las pruebas, uno de ellos con mejores condiciones de hardware para comparar los resultados de ambos:

- **Dispositivo 1** Este dispositivo es sobre el que se ha desarrollado la aplicación móvil, se trata de un dispositivo Android con versión del SO 6.0, 5,2 pulgadas, una resolución de 1920 x 1080, una relación de aspecto de 16:9 y Bluetooth 4. En cuanto a conectividad se dispone de Bluetooth 4.1 y antenas GPS y GLONASS.
- **Dispositivo 2** Este dispositivo se ha utilizado exclusivamente para hacer pruebas de la aplicación una vez terminado el desarrollo. Este dispositivo es un smartphone con versión de Android 9.0, 6,4 pulgadas, una resolución de 2400 x 1080 y una relación de aspecto de 20:9. En cuanto a conectividad este dispositivo dispone de Bluetooth 5.0 y antenas GPS, GLONASS y Galileo.

6.1. Diseño responsive

La primera batería de pruebas se ha centrado en comprobar la adaptabilidad de las pantallas a los diferentes dispositivos. El diseño ha tenido en cuenta que la aplicación puede ser utilizada por diferentes tamaños de pantalla, resoluciones y formatos por lo que se debía hacer un desarrollo capaz de adaptarse a todo tipo de pantallas.

En Figura 6.1 se muestran las dos pantallas diseñadas en ambos dispositivos. En la parte superior se sitúan las pruebas realizadas con el Dispositivo 1, bajo ellas están las del Dispositivo 2. La principal diferencia entre ambos dispositivos es la relación de aspecto. Ambos dispositivos tienen un ancho de pantalla similar, pero el Dispositivo 2 tiene un alto notablemente mayor que el Dispositivo 1. En ambas pantallas la disposición de los botones y textos es correcta, se ha de destacar que para el Dispositivo 2 el mapa es de mayor tamaño lo que supone una ventaja para el usuario, de cualquier modo el usuario puede ampliar o reducir el zoom del mapa para visualizar con mayor o menor detalle la ruta recorrida.

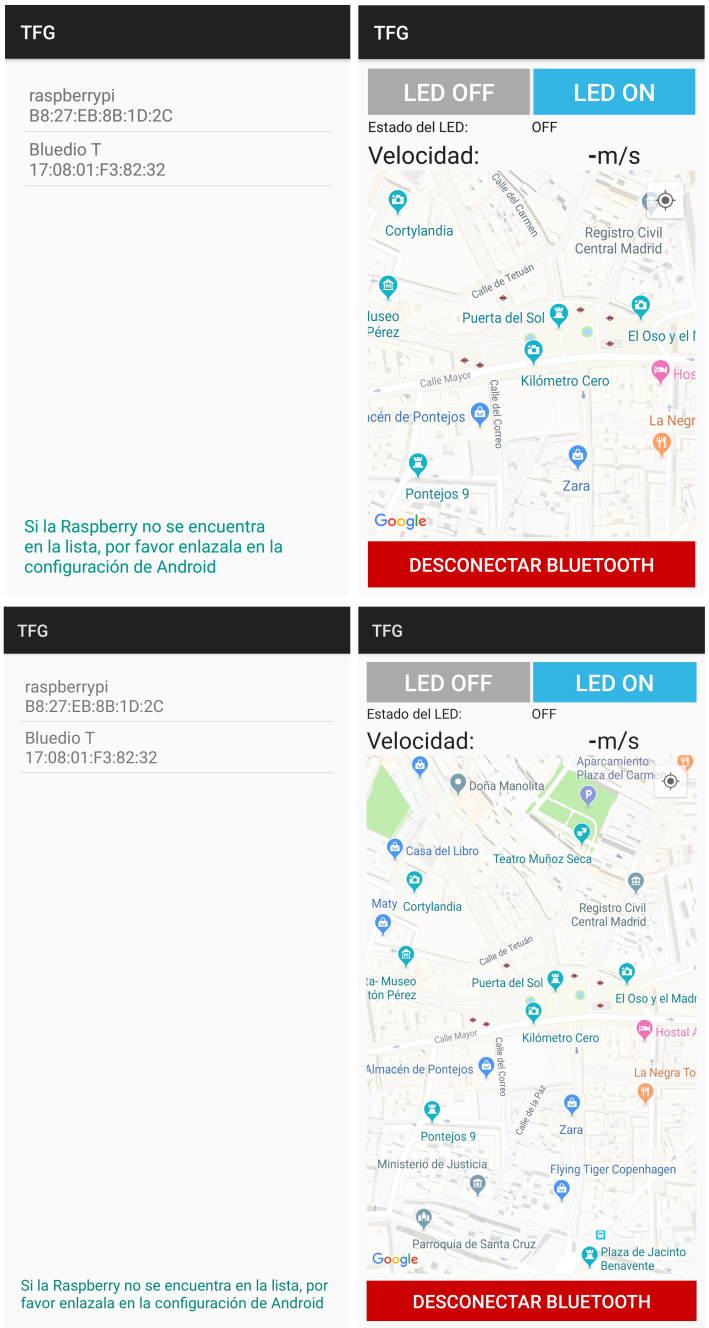


Fig. 6.1. Pantallas en el Dispositivo 1 y en el Dispositivo 2

Android Studio (el SDK oficial para desarrollar aplicaciones Android) incluye una opción para visualizar las pantallas diseñadas en diferentes tamaños y escalas. Se han hecho pruebas con esta herramienta y no se han observado problemas con ningún tipo de dispositivo. La pantalla mas grande que dispone esta herramienta es la de un Nexus 10 que tiene una pantalla de 10,1 pulgadas, y la más pequeña la de un Nexus One con una pantalla de 3,7 pulgadas (ver Figura 6.2).

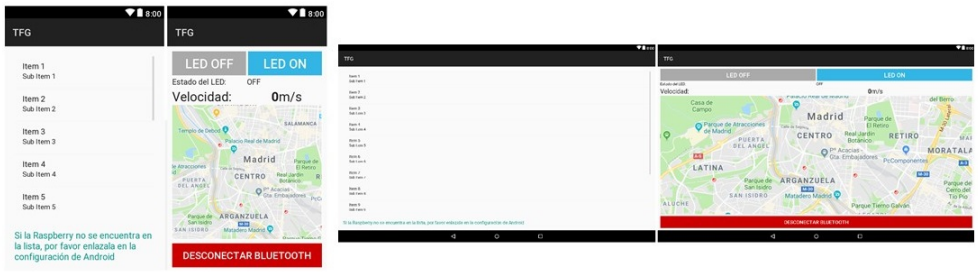


Fig. 6.2. Pantallas en un Nexus One y en un Nexus 10

Otros resultados atener en cuenta es que en pantallas de tabletas o PCs (pantallas de gran tamaño) la aplicación no tiene el formato óptimo. Esta no es una gran problemática ya que la aplicación esta orientada a smartphone y estos no suelen superar las 7 pulgadas, pero aún así se plantea, como una posible mejora, un diseño alternativo para pantallas de gran tamaño.

6.2. Conectividad Bluetooth y GPS

En cuanto a la conexión Bluetooth no se han observado diferencias apreciables entre ambos dispositivos. El alcance en ambos dispositivos es similar y la velocidad de conexión tampoco destaca uno sobre el otro. Ninguno de los dispositivos ha sufrido perdidas de conexión ni retardos en el envío o recepción de los datos.

En cuanto a la los datos recibidos por GPS se han apreciado diferencias destacables entre los dispositivos. En la Tabla 6.1 se puede observar que el tiempo que tarda en obtener la posición el Dispositivo 1 en interiores es casi tres veces mayor al tiempo que tarda el Dispositivo 2. No se puede saber con certeza si esto es debido a que el hardware el mejor en el Dispositivo 2 (procesador, ram, etc.) o si es porque este tiene acceso a la información proporcionada por los satélites de Galileo, cosa que el Dispositivo 1 no tiene.

Dispositivo	Zona de prueba	Tiempo de obtención de la primera posición	Tiempo de obtención de la primera velocidad
1	Interiores	14s	>14s
	Exteriores	<1s	<1s
2	Interior	5s	>5s
	Exteriores	<1s	<1s

TABLA 6.1. TIEMPO EN OBTENCIÓN DE DATOS VIA GPS

En varias pruebas se ha apreciado que ambos dispositivos sufren perdidas de conexión similares cuando se entra en zonas subterráneas o por túneles largos.

Se puede concluir que la conexión Bluetooth es robusta en ambos dispositivos, disponiendo la Raspberry Pi de Bluetooth 4.2, el Dispositivo 1 Bluetooth 4.1 y el Dispositivo 2 Bluetooth 5.0. También que la conexión GPS es mas rápida para el Dispositivo 2, que dispone de GPS, GLONASS y Galileo que para el Dispositivo 1 el cual solo cuenta con GPS y GLONASS. Ambos dispositivos obtienen peores datos en interiores y sufren perdidas de conexión similares al entrar en túneles o zonas subterráneas.

6.3. Posicionamiento a velocidad nula

En cuanto al posicionamiento el Dispositivo 1 cuenta con antenas GPS y GLONASS mientras que el Dispositivo 2 dispone a demás de Galileo. Para el posicionamiento, el dispositivo además de utilizar estos satélites también emplea la información proporcionada por los datos móviles y/o Wifi al que se encuentra conectado.

Ambos dispositivos son capaces de proporcionar una posición parecida a la real con margenes de error de aproximadamente cinco metros. En Figura 6.3 se aprecia la posición obtenida en ambos dispositivos en exteriores y con Wifi y datos móviles activados, los dos obtienen resultados similares en cuanto a precisión pero el Dispositivo 2 destaca por la velocidad de obtención del dato. Esto, a demás de poder ser debido a los satélites Galileo a los que tiene acceso el Dispositivo 2, también puede ser debido a que el procesador y la RAM del Dispositivo 2 permiten un rendimiento mejor que el del Dispositivo 1. No se puede saber con certeza cual de las dos situaciones es la real, pero se podría resolver utilizando dos dispositivos exactamente iguales en cuanto a especificaciones uno de ellos con GPS y GLONASS y otro con GPS, GLONASS y Galileo, en este caso no se dispone de dispositivos con estas características.

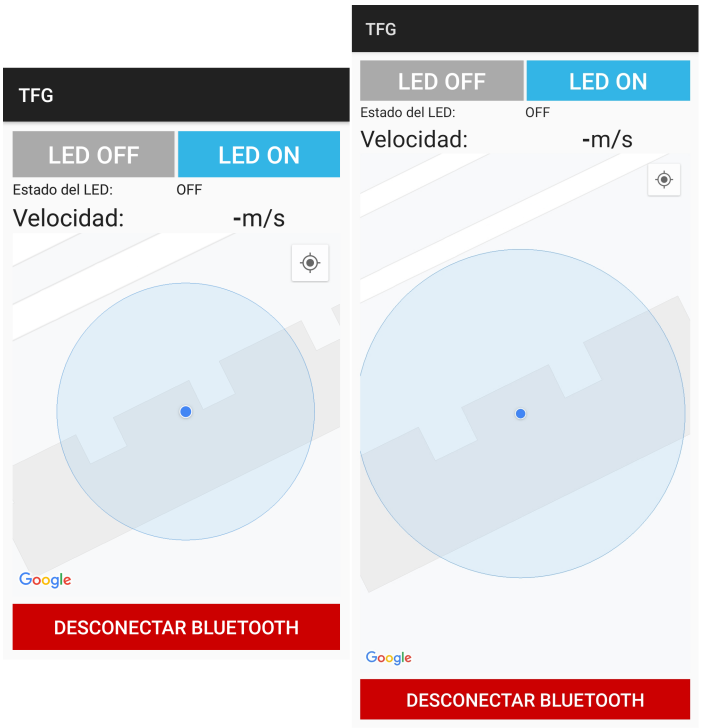


Fig. 6.3. Primera ubicación obtenida tras la primera conexión

Para evaluar cual de los dos obtiene mejor dato de ubicación ante varias casuisticas se ha procedido a realizar tres pruebas con ambos dispositivos:

- Prueba 1: En el primer caso de prueba se deja el dispositivo en una posición fija durante un minuto en exteriores. El GPS se encuentra activado al igual que los datos móviles y el Wifi.
- Prueba 2: En el segundo caso de prueba se deja el dispositivo en una posición fija durante un minuto en exteriores. El GPS se encuentra activado y los datos móviles y el Wifi desactivados.
- Prueba 3: En el tercer caso de prueba se deja el dispositivo en una posición fija durante un minuto en interiores. El GPS se encuentra activado al igual que los datos móviles y el Wifi.

Prueba 1

Este caso es el que dispone de las condiciones más optimas para la obtención de una posición. Cuando la velocidad es nula se observa que para el Dispositivo 2 la precisión es mayor que la del Dispositivo 1 (ver Figura 6.4). La marca roja sobre el mapa representa la ubicación real aproximada, se puede observar que el Dispositivo 2 obtiene puntos más alejados por lo que traza lineas más grandes y dispersas pero con el paso del tiempo consigue una ubicación mas parecida a la real.

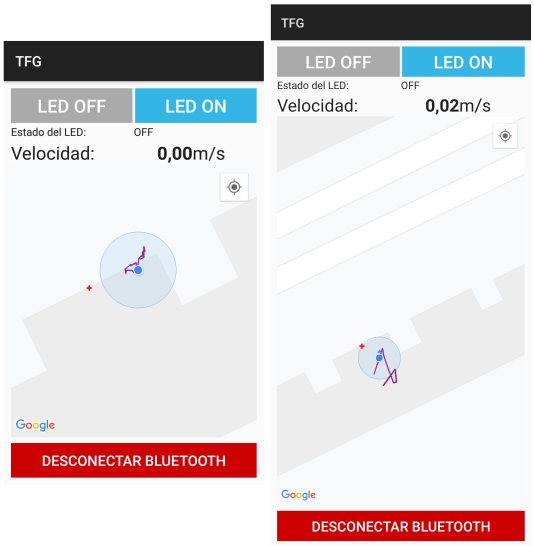


Fig. 6.4. Primera prueba: Exteriores, GPS, Wifi y Datos móviles

Prueba 2

Esta prueba se deshabilitan los datos móviles y el Wifi, estos métodos permiten mejorar los datos de ubicación, desconectandolos los resultados obtenidos son peores a los de la Prueba 1. En Figura 6.5 la marca roja sobre el mapa es la ubicación real aproximada, se puede determinar que a pesar de que la ubicación inicial del Dispositivo 1 es mas precisa este tiene mas dificultades para obtener un dato de ubicación que el Dispositivo 2.

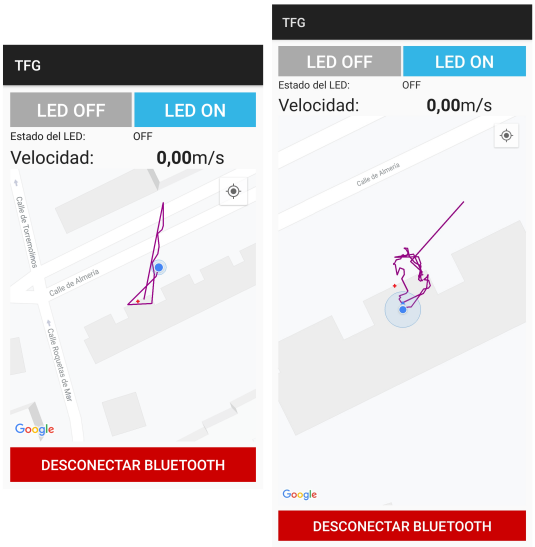


Fig. 6.5. Segunda prueba: Exteriores solo GPS

Prueba 3

En la ultima prueba realizada se ha procurado dificultar lo máximo posible la recepción de datos GPS pero manteniendo la conexión de datos móviles y Wifi. Los resultados obtenidos en esta casuistica son los peores de las tres pruebas realizadas. En la Figura 6.6 se muestra una marca roja sobre el mapa que indica la posición des dispositivo. El Dispositivo 1 muestra un error de varias decenas de metros y no termina de ubicar correctamente la posición. Los resultados del Dispositivo 2 son algo mas prometedores, no se termina de conseguir una ubicación precisa pero con el paso del tiempo el error se va reduciendo.

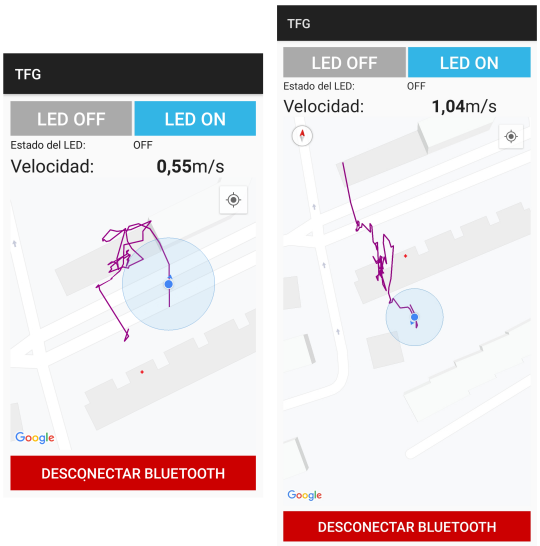


Fig. 6.6. Tercera prueba: Interiores, GPS, Wifi y Datos móviles

6.4. Trazado de ruta y obtención de velocidad

La precisión del dato de velocidad no es fundamental ya que en el proyecto final obtendrá la velocidad en el SIITR (I). Aún así se han prestado atención a los datos obtenidos a la hora de realizar las pruebas de trazado de ruta ya que este método de obtención de la velocidad se puede usar para mejorar la precisión de la obtenida por el SIITR (I) pero no se han realizado medidas de contraste para evaluar la precisión de los datos obtenidos.

Para evaluar la precisión del trazado de la ruta se han realizado tres pruebas con diferentes velocidades con los dos dispositivos para observar el efecto de la velocidad y de las antenas GPS en el trazado de la ruta:

- Prueba 1: En el primer caso de prueba se realizará en tren, la velocidad debería variar entre 70 y 120Km/h.
- Prueba 2: El segundo caso de prueba se llevará a cabo en automóvil por ciudad, el intervalo de velocidad será de entre 20 y 60Km/h.

- Prueba 3: El tercer caso de prueba se recorre una ruta a pie, la velocidad variará entre 0 y 5Km/h.

Prueba 1

Esta prueba será la de mayor velocidad, en Figura 6.7 se observa as rutas captadas por la aplicación móvil. Estas rutas no son totalmente reales ya que el recorrido real es el de las vías del tren dibujadas sobre los mapas. El error de la ruta es de varios metros pero se representa con claridad la ruta recorrida. Se observa que cuando el tren se detiene la aplicación la ruta se vuelve imprecisa hasta que se reanuda el movimiento.

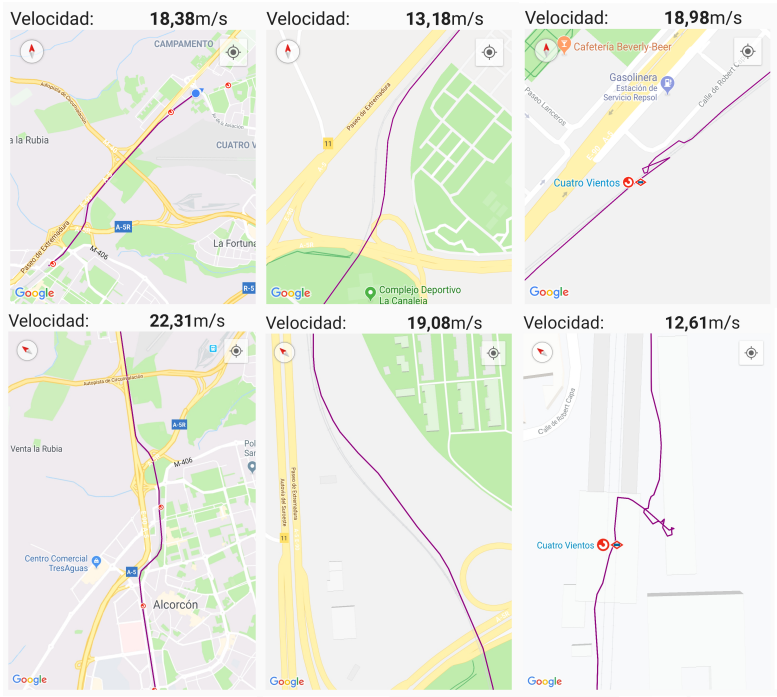


Fig. 6.7. Primera prueba: Ruta en tren

Los resultados para ambos dispositivos son similares en cuanto a precisión, destaca ligeramente el Dispositivo 2, como se muestra en la Figura 6.8 los peores resultados son en curvas a alta velocidad. Los dos dispositivos tienden a seguir una linea recta en vez de realizar la curva, pero el Dispositivo 2 es capaz de recuperar la posición más rápidamente que el Dispositivo 1. Con respecto a la velocidad no se ha detectado ningún valor extraño en las medidas.

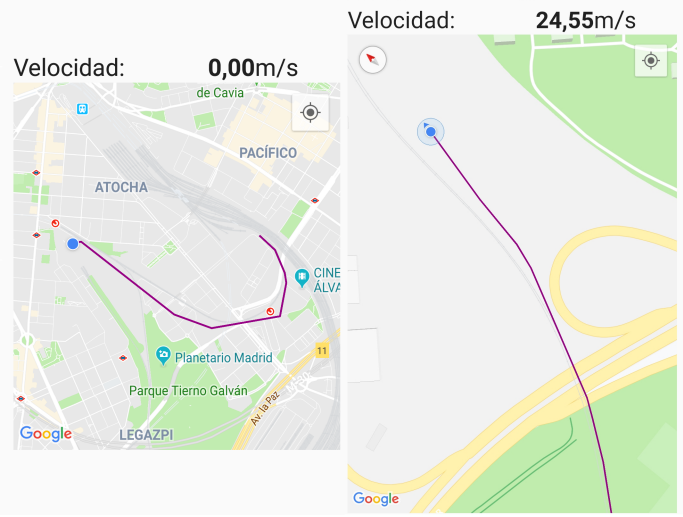


Fig. 6.8. Primera prueba: Error de ruta en tren

Prueba 2

En la Figura 6.9 se muestran varias rutas realizadas en automóvil, la ruta dibujada sobre el mapa es similar a la real. En cuanto a las velocidades obtenidas se encuentran en el rango esperado.

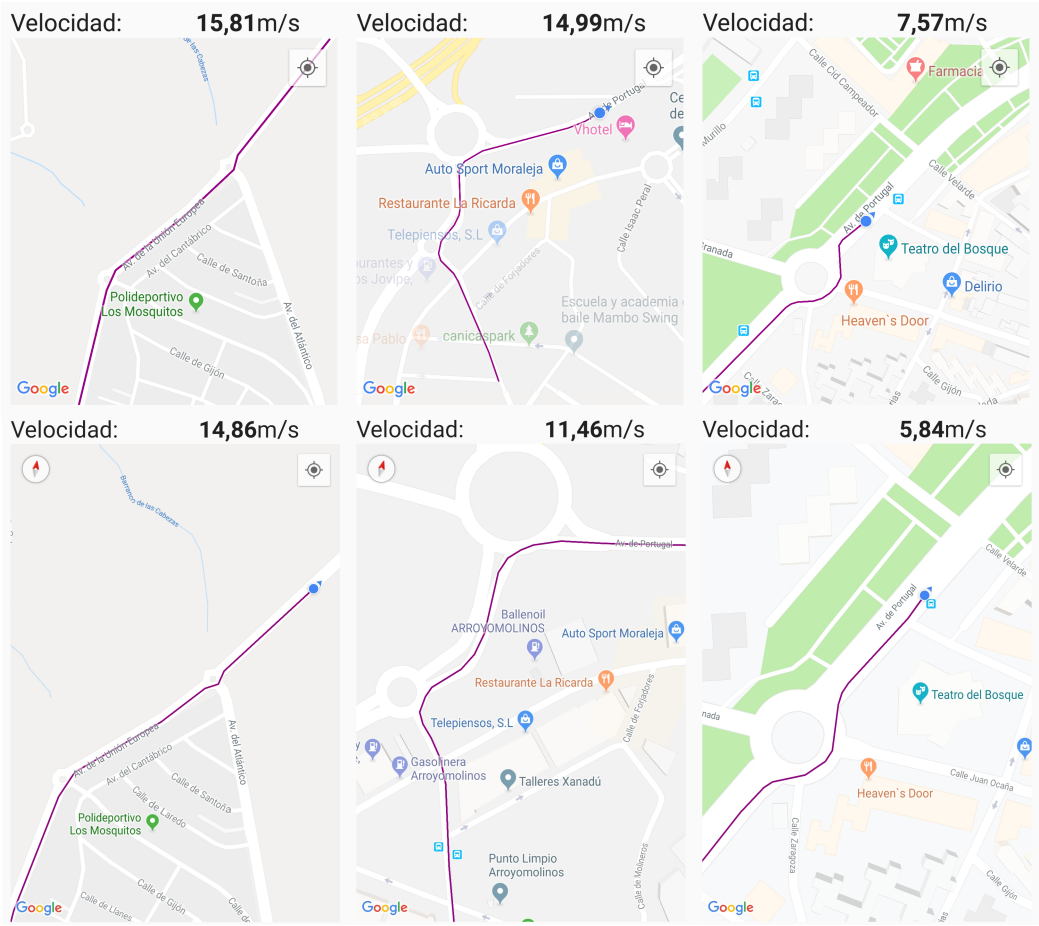


Fig. 6.9. Segunda prueba: Ruta en coche

Acercando la imagen en las glorietas y curvas se puede observar que estas son las zonas mas problemática en el trazado de la ruta, el Dispositivo 2 consigue una ruta ligeramente mas precisa que el Dispositivo 1 en estas zonas (ver Figura 6.10). Esto también depende de la velocidad a la que se realiza la curva, a menor velocidad mayor precisión.

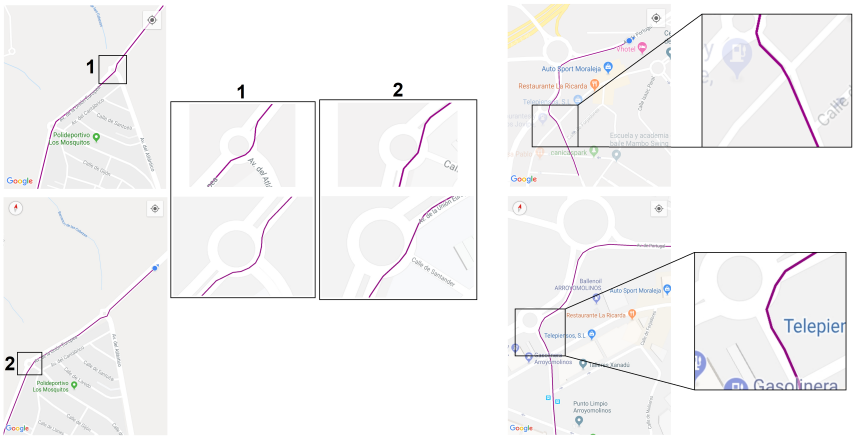


Fig. 6.10. Segunda prueba: Error de ruta en coche

Prueba 3

Esta prueba será la que se realice a velocidades más similares a las esperadas de un monopatín o patinete. En la Figura 6.11 se puede observar la ruta recorrida para ambos dispositivos. La representación de la ruta es igual a la ruta real, el error para ambos dispositivos es de menos de 1 metro. El único error producido es al inicio de la ruta pero este error solo se produce cuando la velocidad es nula y cuando comienza la marcha se corrige.

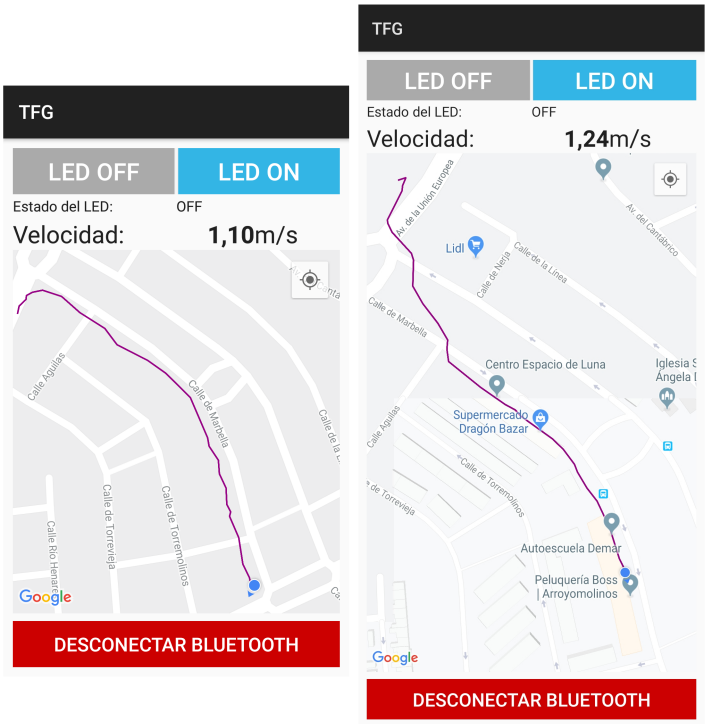


Fig. 6.11. Tercera prueba: Ruta a Pie

Se aprecia que el Dispositivo 2 traza la ruta ligeramente mejor que el Dispositivo 1, la ruta trazada por el Dispositivo 1 sufre ligeros picos mientras la del Dispositivo 2 es más suave.

7. CONCLUSIONES

En esta sección se realizará un análisis de los resultados obtenidos y se evaluará si el sistema desarrollado cumple la funcionalidad esperada y se planteará los siguientes pasos a seguir para completar el desarrollo del sistema.

El proyecto desarrollado en el presente documento ha requerido conocimientos de programación para la implementación de la aplicación Android y para la integración del circuito electrónico y se han requerido conocimientos sobre circuitos electrónicos y protocolos de comunicación Bluetooth. Todo esto ha implicado un aprendizaje que se ha llevado a cabo durante el desarrollo del proyecto y que continuará con los trabajos futuros planeados para este.

Gracias a las pruebas realizadas se ha demostrado que el sistema diseñado es viable para ser utilizado como sistema de navegación y el estudio de de viabilidad de lectura de una PWM demuestra que se podrá integrar el sistema de iluminación sin modificaciones una vez que se finalice su desarrollo para obtener un prototipo final totalmente funcional.

Se ha probado que el trazado de la ruta es muy preciso a bajas velocidades y a altas velocidades la ruta sufre desviaciones muy ligeras con errores máximos de unos 10 metros. El sistema ha sido diseñado para bicicletas, patinetes y monopatines por lo que no surge ninguna problemática derivada de estos resultados ya que estos vehículos no superan los 80Km/h.

La conexión Bluetooth es robusta sin variaciones apreciables entre dispositivos, en cambio la conexión GPS puede sufrir desconexiones en túneles y zonas subterráneas y en interiores la respuesta es mas lenta y menos precisa.

El estudio de viabilidad de utilización de una Rapsberry Pi para la lectura de una PWM ha demostrao que escogiendo el rango de velocidades que el sistema será capaz de leer y la frecuencia de transmisión de la PWM adecuadas se puede utilizar la Raspberry Pi seleccionada para la lectura de la velocidad. Los datos de velocidad obtenidos vía GPS encajan en los resultados esperados pero se ha de realizar un estudio mas profundo sobre la precisión de estos.

7.1. Objetivos cumplidos

Se ha conseguido diseñar una aplicación Android capaz de conectar vía Bluetooth un circuito electrónico y cualquier dispositivo con una versión de Android 4.0. Gracias a la aplicación se podrá controlar la iluminación del circuito electrónico y esta además es capaz obtener la ruta recorrida por el usuario en tiempo real y trazarla sobre un mapa. También se ha conseguido que la aplicación, de manera autónoma, sea capaz de obtener la velocidad del usuario para controlar el circuito y de que manera visual el usuario se

capaz de saber si circula a una velocidad que cumple las normativas vigentes.

7.2. Líneas futuras de trabajo

Como ya se ha comentado anteriormente este proyecto forma parte de uno mas grande que aun se encuentra en desarrollo, esto implica que hasta que no se finalice la otra parte del proyecto se pueden hacer modificaciones en este hasta finalmente integrar ambos creando un prototipo final. Algunos de los trabajos futuros que se realizarán, ordenados por prioridad, son las siguientes:

- Creación de un método de emparejamiento de dispositivos para que no se requiera que se encuentren previamente emparejados.
- Creación de un botón para poder seleccionar el vehículo que se esta utilizando (bicicleta, patinete o monopatín) y de manera dinámica adaptar el limite de velocidad para el encendido y apagado de la iluminación indicadora del limite.
- Pruebas detalladas para medir la precisión de la velocidad obtenida por la aplicación móvil.
- Creación de un método de control del inicio y finalización del trazado de ruta.
- Creación de un sistema de control del funcionamiento de la aplicación móvil y de la aplicación para Raspberry mediante logs para reportarlos en caso de error.
- Búsqueda de microcontrolador para hacer las funciones de la Raspberry (la Raspberry utilizada tiene un coste de 45 euros lo que encarece el sistema).
- Estudio de viabilidad de alimentación autorecargable: dinamo o alimentación mediante paneles solares.
- Integrar las dos partes del proyecto y diseñar etapa de potencia para alimentarlo todo con una batería.
- Estudio sobre puntos de colocación mas óptimos para bicicletas, patinetes y monopatines.
- Integrar todo el sistema en una caja resistente a golpes, al agua y al polvo.
- Diseñar una interfaz de usuario más agradable para la aplicación al igual que un logotipo.
- Crear usuarios para la aplicación para que se puedan seguir entre ellos y puedan compartir sus datos de navegación.

BIBLIOGRAFÍA

[1] "ONU-Hábitat, el programa de las Naciones Unidas para los asentamientos humanos y la urbanización sostenible.", *ONU-Habitat*, [En línea]. Disponible en: http://es.unhabitat.org/temas-urbanos/cambio-climatico (Acceso: 28-04-2019).

[2] J. J. V. Cenarruzabeitia, J. A. M. Hernández y M. á. Martínez-González, "Beneficios de la actividad física y riesgos del sedentarismo", *Medicina clínica*, vol. 121, n.º 17, pp. 665-672, 2003.

[3] S. Márquez, "Beneficios psicológicos de la actividad física", *Revista de psicología general y aplicada: Revista de la Federación Española de Asociaciones de Psicología*, vol. 48, n.º 1, pp. 185-206, 1995.

[4] Á. C. Sánchez, M. F. García y P. D. F. SAGUAR, "Análisis económico y medioambiental del sector eléctrico en España", *Estudios de Economía Aplicada*, vol. 29, n.º 2, pp. 493-514, 2011.

[5] RACE y Universidad Carlos III, "Iluminación del automóvil y seguridad vial", *Estudio RACE*, pp. 4-71, 2006.

[6] M. Tixce, "El origen del faro y las luces de freno", *Motor y Racing*, [En línea]. Disponible en: www.motoryracing.com/coches/noticias/el-origen-del-faro-y-las-luces-de-freno/ (Acceso: 15-05-2019).

[7] "Bicicleta", *Rasgando la Oscuridad, Alumbrado en el Trasporte Terrestre*, 2014. [En línea]. Disponible en: http://www.rasgandolaoscuridadtransporteterrestr.blogspot.com/2014/12/bicicleta.html (Acceso: 16-05-2019).

[8] "Sistemas de iluminación para bicicletas", *Terra Ecología práctica*, [En línea]. Disponible en: http://www.terra.org/categorias/comunidad-ecotransporte/sistemas-de-iluminacion-para-bicicletas (Acceso: 15-05-2019).

[9] M. Déleg y A. E. Cuenca, "Tecnología Led", *Electrónica digital*,

[10] J. M. López-Guillén, "Ciclistas protegidos con luz:un aragonés inventa un sistema de seguridad para bicicletas", *20 Minutos*, 2011. [En línea]. Disponible en: http://www.20minutos.es/noticia/973217/0/ciclistas/protegidos/luz/ (Acceso: 17-05-2019).

[11] M. P. Standley, *Skateboard having lighting system*, US Patent 5,067,058, nov. de 1991.

[12] J. P. Kertes, *Illuminated scooter*, US Patent 7,311,164, dic. de 2007.

[13] J. Fallas, "Sistema de posicionamiento global", *Costa Rica: Universidad Nacional Heredia*, p. 10, 2010.

[14] A. Pozo-Ruz et al., “Sistema de posicionamiento global (GPS): Descripción, análisis de errores, aplicaciones y futuro”, *ETS ingenieros de Telecomunicaciones, Universidad de Malaga*, 2000.

[15] D. A. G. Álvarez, "Sistema GNSS (global navigation satellite system)", Trabajo fin de grado, Dpto. de Ingeniería Informática, Universidad Autónoma de Madrid, Madrid, España, 2008.

[16] L. Letham, *GPS fácil. Uso del sistema de posicionamiento global*. Editorial Paidotribo, 2001, vol. 67.

[17] L. Gauthier, P. Michel, J. Ventura-Traveset y J. Benedicto, “EGNOS: the first step in Europe’s contribution to the global navigation satellite system”, *ESA bulletin*, vol. 105, pp. 35-42, 2001.

[18] “Galileo, ya disponible para el mundo”, *European Space Agency*, dic. de 2016. [En línea]. Disponible en: `http://www.esa.int/esl/ESA_in_your_country/Spain/Galileo_ya_disponible_para_el_mundo` (Acceso: 18-05-2019).

[19] D. Gutiérrez, "Sistema pasarela Bluetooth para una red de sensores Zigbee", Trabajo fin de grado, Dpto. de Ingeniería Electrónica, Universidad de Sevilla, Andalucía, España, 2009.

[20] A. Pothitos, “The History of Bluetooth”, *Mobile Industry Review*, 2017. [En línea]. Disponible en: `http://www.mobileindustryreview.com/2017/08/the-history-of-bluetooth.html` (Acceso: 28-04-2019).

[21] J. J. M. Durá et al., “Utilización de sensores Bluetooth en la monitorización de datos de tráfico”, *Cuadernos Tecnológicos de la PTC.*, vol. Nº 7, 2015.

[22] C. L. Róvere, J. E. Plaza, W. Silva, A. Urbano y V. Utrera, “Bluetooth”, *Dpto. de Redes de Computadoras II, Universidad Simón Bolívar*, [En línea]. Disponible en: `http://www.ldc.usb.ve/~poc/RedesII/Grupos/G1` (Acceso: 28-04-2019).

[23] B. Mitchell, “Introduction to MAC Addresses With Formatting Examples”, *Lifewire*, 2018. [En línea]. Disponible en: `http://www.lifewire.com/introduction-to-mac-addresses-817937` (Acceso: 28-04-2019).

[24] M. Gargenta, *Learning android*. O’Reilly Media, Inc., 2011.

[25] A. P. Intriago, "Diseño e implementación de un sistema de monitorización de temperatura capaz de comunicar de manera inalámbrica con un dispositivo móvil", Trabajo fin de grado, Dpto. de Ingeniería Electrónica, Universidad Carlos III, Madrid, España, 2014.

[26] L. Llamas, “¿Qué es una FPGA? Motivos de su auge en la comunidad maker”, *Luis Llamas Ingeniería, informática y diseño*, 2017. [En línea]. Disponible en: `http://www.luisllamas.es/que-es-una-fpga/` (Acceso: 23-05-2019).

[27] "¿Qué es una FPGA? Motivos de su auge en la comunidad maker", *EducacionIT*, 2018. [En línea]. Disponible en: http://www.blog.educacionit.com/2018/07/10/java-vs-python-que-lenguaje-de-programacion-es-mejor-para-ti/ (Acceso: 24-05-2019).

[28] "Usage-GPIO", *Raspberry Pi Foundation*, [En línea]. Disponible en: http://www.raspberrypi.org/documentation/usage/gpio/ (Acceso: 25-05-2019).

[29] "Hardware-GPIO", *Raspberry Pi Foundation*, [En línea]. Disponible en: http://www.raspberrypi.org/documentation/hardware/raspberrypi/gpio/ (Acceso: 25-05-2019).

[30] "GPIO Electrical Specifications, Raspberry Pi Input and Output Pin Voltage and Current Capability", *Mosaic Documentation Web*, [En línea]. Disponible en: http://www.mosaic-industries.com/embedded-systems/microcontroller-projects/raspberry-pi/gpio-pin-electrical-specifications (Acceso: 25-05-2019).

[31] "Bluetooth overview", *Android Developers*, [En línea]. Disponible en: https://developer.android.com/guide/topics/connectivity/bluetooth.html (Acceso: 26-05-2019).

[32] N. Jennings, "Bluetooth overview", *Real Python*, 2018. [En línea]. Disponible en: https://realpython.com/python-sockets/ (Acceso: 27-05-2019).

[33] R. Madriaga, "Socket Programming in Python (Guide)", *Curso Android Studio*, 2015. [En línea]. Disponible en: https://cursoandroidstudio.blogspot.com/2015/10/conexion-bluetooth-android-con-arduino.html (Acceso: 01-06-2019).

ANEXO A. CÓDIGO PARA RASPBERRY

CÓDIGO 1. Control de LEDs

```
import bluetooth
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)
GPIO.setup(3, GPIO.OUT)# Blue LED GPIO3 as an output
GPIO.setup(5, GPIO.OUT)# Green LED GPIO5 as an output
GPIO.setup(6, GPIO.OUT)# Red LED GPIO6 as an output

# Initialize GPIO
GPIO.output(3, 0)
GPIO.output(5, 0)
GPIO.output(6, 0)

# Print "start" to inform that the app started
print("Start\n\n")

# Functions
def Leds():
        while True:
                data = client.recv(1024).decode()
                print("Revived: %s" % data)
                if data == "0":
                        GPIO.output(3, 0)
                        print("LED off")
                if data == "1":
                        GPIO.output(3, 1)
                        print("LED on")
                if data == "f":
                        GPIO.output(5, 1)
                        GPIO.output(6, 0)
                if data == "t":
                        GPIO.output(5, 0)
                        GPIO.output(6, 1)
                if data == "q":
                        # A "q" means that the user wants to finish
                            the Bt connection so
                        # we'll turn down all the LEDs and finish in
                             the main loop
                        print("Bluetooth Disconnected\n\n")
                        GPIO.output(3, 0)
                        GPIO.output(5, 0)
                        GPIO.output(6, 0)
                        break
```

CÓDIGO 2. Conexión Bluetooth

```
# Main Loop
while True:
        server = bluetooth.BluetoothSocket(bluetooth.RFCOMM) # Create RFCOMM socket
        server.bind(("", 0)) # Bind the socket to all ports
        server.listen(1) # Listen for a connection

        try:
                client, address = server.accept() # Accept a connection from a client with its address
                print("Accepted connection from %s" % str(address)) # Print the address of the connected device
                Leds() # Start receive loop

        except bluetooth.BluetoothError as e:
                print("Bluetooth not connected: [%s]\n\n" % str(e)) # If connection failed print the error

        # Close client and socket
        client.close()
        server.close()
```

ANEXO B. CÓDIGO APLICACIÓN ANDROID

CÓDIGO 3. Código de obtención de dispositivos emparejados[33]

```
// Get the local Bluetooth adapter
mBtAdapter = BluetoothAdapter.getDefaultAdapter();

// Get a set of currently paired devices and append to '
   pairedDevices'
Set<BluetoothDevice> pairedDevices = mBtAdapter.getBondedDevices();

// Add previosuly paired devices to the array
if (pairedDevices.size() > 0) {
        for (BluetoothDevice device : pairedDevices) {
                mPairedDevicesArrayAdapter.add(device.getName() + "\
                   n" + device.getAddress());
        }
} else {
        String noDevices = getResources().getText(R.string.
           none_paired).toString();
        mPairedDevicesArrayAdapter.add(noDevices);
}
```

CÓDIGO 4. Código de seleccion de un dispositivo[33]

```
private OnItemClickListener mDeviceClickListener = new
   OnItemClickListener() {
        public void onItemClick(AdapterView<?> av, View v, int arg2,
            long arg3) {

                textView1.setText("Conectando...");
                // Get the device MAC address, which is the last 17
                   chars in the View
                String info = ((TextView) v).getText().toString();
                String address = info.substring(info.length() - 17);

                // Make an intent to start next activity while
                   taking an extra which is the MAC address.
                Intent i = new Intent(DeviceListActivity.this,
                   MainActivity.class);
                i.putExtra(EXTRA_DEVICE_ADDRESS, address);
                startActivity(i);
        }
};
```

CÓDIGO 5. Creación del Socket[33]

```
private BluetoothSocket createBluetoothSocket(BluetoothDevice device
    ) throws IOException {
        try {
                Method method;

                method = device.getClass().getMethod("
                    createRfcommSocket", new Class[]{int.class});
                return (BluetoothSocket) method.invoke(device, 1);
        } catch (Exception e) {
                System.out.println("Error creating socket: " + e);
                System.out.println("Will try with another kind of
                    socket");
                try {
                        return device.
                            createRfcommSocketToServiceRecord(
                            BTMODULEUUID);
                } catch (Exception e2) {
                        System.out.println("Error creating second
                            socket: " + e2);
                        return null;
                }
        }
}
```

CÓDIGO 6. Conexión Bluetooth[33]

```
@Override
public void onResume() {
        super.onResume();

        //Get MAC address from DeviceListActivity via intent
        Intent intent = getIntent();

        //Get the MAC address from the DeviceListActivty via EXTRA
        address = intent.getStringExtra(DeviceListActivity.
            EXTRA_DEVICE_ADDRESS);

        //create device and set the MAC address
        BluetoothDevice device = btAdapter.getRemoteDevice(address);
        System.out.println("Device created");
        try {
                btSocket = createBluetoothSocket(device);
                System.out.println("Socket created");
        } catch (IOException e) {
                System.out.println("Socket creation failed");
                System.out.println("Error: " + e);
        }
        // Establish the Bluetooth socket connection.
        try {
```

```
                btSocket.connect();
                System.out.println("Connected");
        } catch (IOException e1) {
                System.out.println(e1);
                try {
                        btSocket.close();
                } catch (IOException e2) {
                        System.out.println(e2);
                }
        }
        mConnectedThread = new ConnectedThread(btSocket);
        mConnectedThread.start();

        //I send a character when resuming.beginning transmission to
            check device is connected
        //If it is not an exception will be thrown in the write
           method and finish() will be called
        mConnectedThread.write("f");  //"f" means users speed its
           not above limit
}
```

CÓDIGO 7. Comprobación de Bluetooth[33]

```
private void checkBTState() {
        // Check device has Bluetooth and that it is turned on
        mBtAdapter=BluetoothAdapter.getDefaultAdapter();
        if(mBtAdapter==null) {
                //A null default adapter means that the device does
                   not have Bluetooth
                System.out.println("This device doesn't have
                   Bluetooth");
        } else {
                if (mBtAdapter.isEnabled()) {
                } else {
                        //Prompt user to turn on Bluetooth
                        Intent enableBtIntent = new Intent(
                           BluetoothAdapter.ACTION_REQUEST_ENABLE);
                        startActivityForResult(enableBtIntent, 1);

                }
        }
}
```

CÓDIGO 8. Comprobación de GPS[33]

```
private void checkGpsState() {
        locManager = (LocationManager) this.getSystemService(Context
           .LOCATION_SERVICE);
        if (locManager.isProviderEnabled(LocationManager.
           GPS_PROVIDER)) {
```

```
        } else {
                AlertDialog.Builder constructor = new AlertDialog.
                    Builder(this);
                constructor.setCancelable(false);
                constructor.setTitle("GPS desactivado");
                constructor.setMessage("El GPS esta desactivado, es
                    necesario el GPS para obtener la velocidad");
                constructor.setPositiveButton("Ajustes", new
                    DialogInterface.OnClickListener() {
                     @Override
                     public void onClick(DialogInterface dialog,
                         int which) {

                          Intent enableGps = new Intent(
                              android.provider.Settings.
                              ACTION_LOCATION_SOURCE_SETTINGS)
                              ;
                          startActivityForResult(enableGps, 1)
                              ;
                     }
                });
                constructor.show();
        }
}
```

CÓDIGO 9. Creación del objeto tipo LocationManager

```
LocationManager locManager = null;

[...]

locManager = (LocationManager) this.getSystemService(Context.
    LOCATION_SERVICE);

[...]

locManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 500,
     0, this);
```

CÓDIGO 10. Obtención de la posición y velocidad

```
@Override
public void onLocationChanged(Location location) {
        if (location == null) {
                currentSpeed.setText("-");
        } else {
                //update speed
                speed = location.getSpeed();
                currentSpeed.setText(String.format("%.2f", speed));
                    //print speed with 2 decimals
```

```
9
10                  if (speed > 2.7 && !speedState) { // 2.7m/s equals
                        10Km/h
11                          mConnectedThread.write("t");
12                          speedState = true;
13                  } else if (speedState) {
14                          mConnectedThread.write("f");
15                          speedState = false;
16                  }
17                  System.out.println("Speed is " + speed + "
                        speedState is " + speedState);
18
19                  //update map
20                  currentposition = new LatLng(location.getLatitude(),
                         location.getLongitude());
21                  mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(
                        currentposition, 17));
22                  updateTrack(currentposition);
23          }
24  }
```

CÓDIGO 11. Actualización del mapa

```
1   private void updateTrack(LatLng lastKnownLatLng) {
2           List<LatLng> points = gpsTrack.getPoints();
3           points.add(lastKnownLatLng);
4           gpsTrack.setPoints(points);
5   }
```

CÓDIGO 12. Pantalla 1: Lista de dispositivos

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <LinearLayout xmlns:android="http://schemas.android.com/apk/res/
        android"
3           android:layout_width="match_parent"
4           android:layout_height="match_parent"
5           android:layout_margin="20dp">
6
7           <ListView android:id="@+id/paired_devices"
8                   android:layout_width="match_parent"
9                   android:layout_height="wrap_content"
10                  android:stackFromBottom="false"
11                  android:layout_weight="1" />
12
13          <TextView
14                  android:id="@+id/connecting"
15                  android:layout_width="wrap_content"
16                  android:layout_height="wrap_content"
17                  android:textAppearance="?android:attr/
                        textAppearanceLarge" />
```

```
18
19          <TextView
20                  android:id="@+id/infoText"
21                  android:layout_width="wrap_content"
22                  android:layout_height="wrap_content"
23                  android:text="Si la Raspberry no se encuentra en la
                        lista, por favor enlazala en la configuracion de
                         Android"
24                  android:textAppearance="?android:attr/
                        textAppearanceLarge"
25                  android:textColor="?android:attr/colorAccent"
26                  android:textSize="18dp" />
27
28  </LinearLayout>
```

CÓDIGO 13. Pantalla 2: Mapa

```
1   <LinearLayout xmlns:android="http://schemas.android.com/apk/res/
        android"
2           xmlns:tools="http://schemas.android.com/tools"
3           android:layout_width="match_parent"
4           android:layout_height="match_parent"
5           android:layout_margin="10dp"
6           android:orientation="vertical">
7
8           <LinearLayout
9                   android:layout_width="match_parent"
10                  android:layout_height="wrap_content"
11                  android:orientation="horizontal">
12
13                  <Button
14                          android:id="@+id/buttonOff"
15                          android:layout_width="match_parent"
16                          android:layout_height="wrap_content"
17                          android:layout_marginRight="2dp"
18                          android:layout_weight="1"
19                          android:background="@android:color/
                                darker_gray"
20                          android:text="LED OFF"
21                          android:textColor="@android:color/white"
22                          android:textSize="25sp" />
23
24                  <Button
25                          android:id="@+id/buttonOn"
26                          android:layout_width="match_parent"
27                          android:layout_height="wrap_content"
28                          android:layout_marginLeft="2dp"
29                          android:layout_weight="1"
30                          android:background="@android:color/
                                holo_blue_light"
```

```
31                          android:text="LED ON"
32                          android:textColor="@android:color/white"
33                          android:textSize="25sp" />
34
35          </LinearLayout>
36
37          <LinearLayout
38                  android:layout_width="match_parent"
39                  android:layout_height="wrap_content"
40                  android:layout_marginTop="2dp"
41                  android:orientation="horizontal">
42
43                  <TextView
44                          android:id="@+id/ledStatusTitle"
45                          android:layout_width="match_parent"
46                          android:layout_height="wrap_content"
47                          android:layout_weight="1"
48                          android:text="Estado del LED:"
49                          android:textAppearance="?android:attr/
                               textAppearanceLarge"
50                          android:textSize="15sp" />
51
52                  <TextView
53                          android:id="@+id/ledStatus"
54                          android:layout_width="match_parent"
55                          android:layout_height="wrap_content"
56                          android:layout_weight="1"
57                          android:text="OFF"
58                          android:textAppearance="?android:attr/
                               textAppearanceLarge"
59                          android:textSize="15sp" />
60          </LinearLayout>
61
62          <LinearLayout
63                  android:layout_width="match_parent"
64                  android:layout_height="wrap_content"
65                  android:layout_marginTop="2dp"
66                  android:orientation="horizontal">
67
68                  <TextView
69                          android:id="@+id/textView1"
70                          android:layout_width="wrap_content"
71                          android:layout_height="wrap_content"
72                          android:layout_weight="1"
73                          android:text="@string/velocidad"
74                          android:textAppearance="?android:attr/
                               textAppearanceLarge"
75                          android:textSize="25sp" />
76
77                  <TextView
78                          android:id="@+id/speed"
```

```
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:gravity="right"
                android:text="0"
                android:textAppearance="?android:attr/
                    textAppearanceLarge"
                android:textSize="25sp"
                android:textStyle="bold" />

        <TextView
                android:id="@+id/units1"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:text="m/s"
                android:textAppearance="?android:attr/
                    textAppearanceLarge"
                android:textSize="25sp" />

    </LinearLayout>

    <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:orientation="vertical">

            <fragment
                android:id="@+id/map"
                android:name="com.google.android.gms.maps.
                    SupportMapFragment"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:layout_weight="1"
                tools:context=".activity_main" />

            <Button
                android:id="@+id/ButtonDisconnect"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_alignParentBottom="true"
                android:layout_marginTop="4dp"
                android:background="@android:color/
                    holo_red_dark"
                android:text="Desconectar bluetooth"
                android:textColor="@android:color/white"
                android:textSize="20sp" />
    </LinearLayout>
</LinearLayout>
```

ANEXO C. PRUEBAS DE LECTURA DE SEÑALES PWM

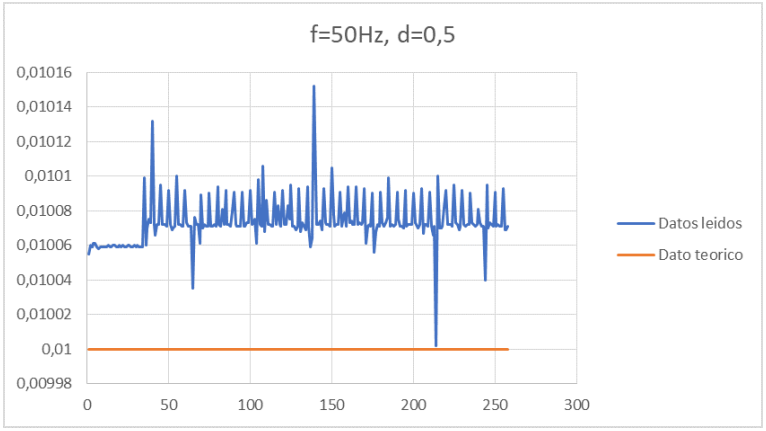


Fig. 1. Lectura de PWM con frecuencia 50Hz y ciclo de trabajo 0,5

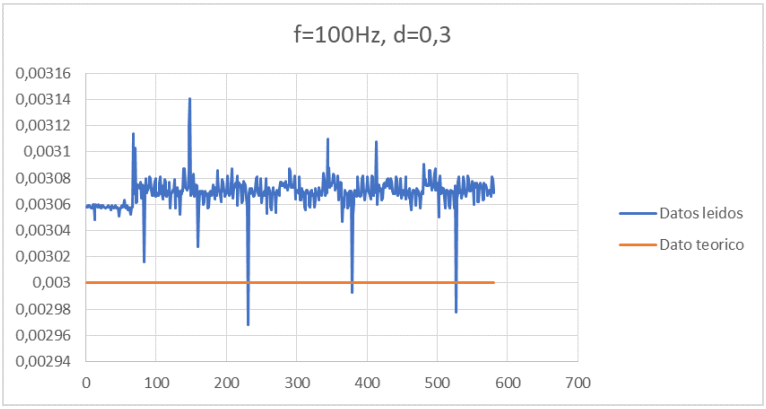


Fig. 2. Lectura de PWM con frecuencia 100Hz y ciclo de trabajo 0,3

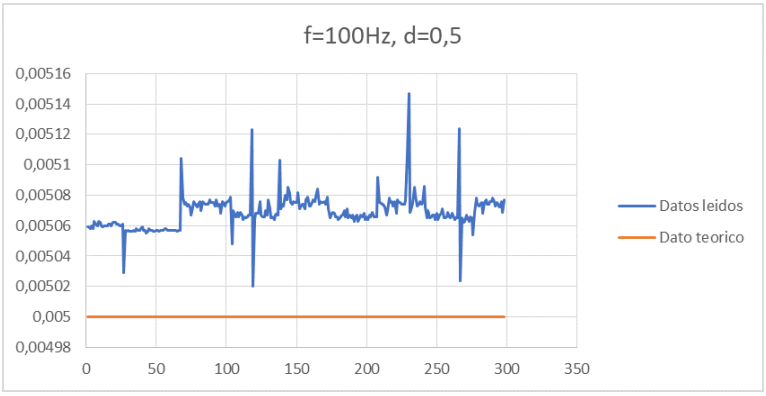


Fig. 3. Lectura de PWM con frecuencia 100Hz y ciclo de trabajo 0,5

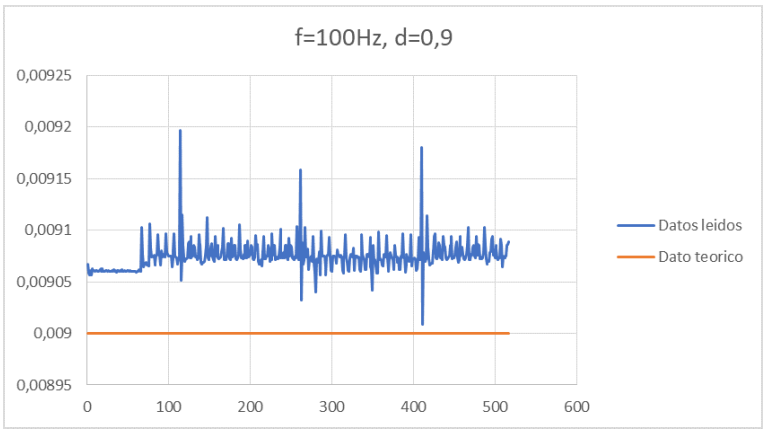


Fig. 4. Lectura de PWM con frecuencia 100Hz y ciclo de trabajo 0,9

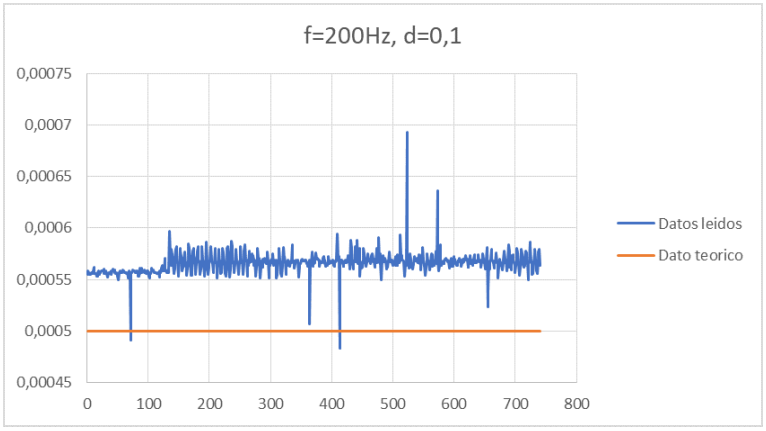


Fig. 5. Lectura de PWM con frecuencia 200Hz y ciclo de trabajo 0,1

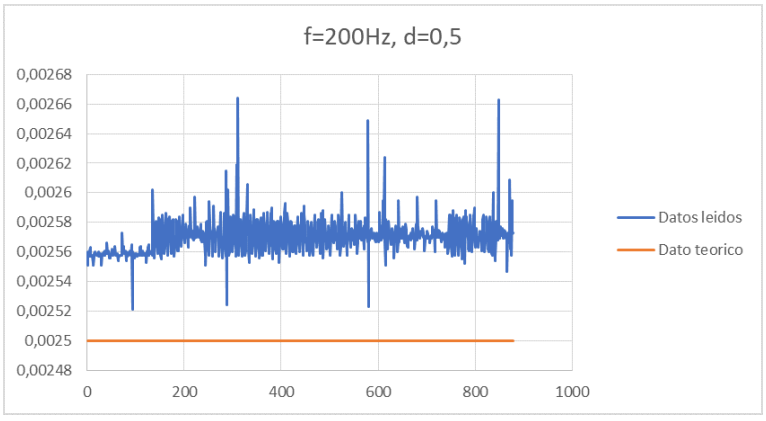


Fig. 6. Lectura de PWM con frecuencia 200Hz y ciclo de trabajo 0,5

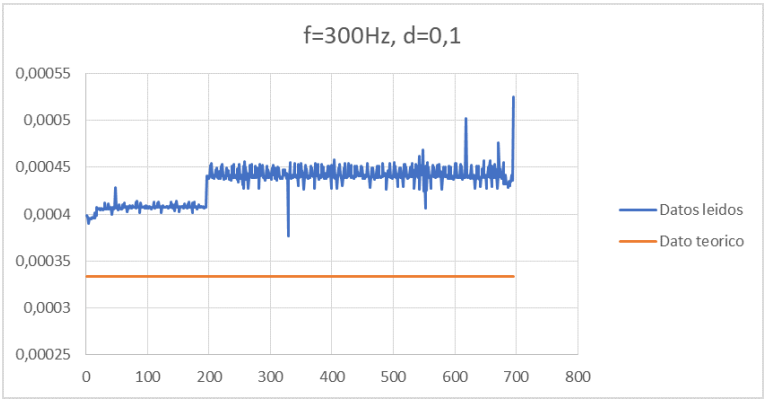


Fig. 7. Lectura de PWM con frecuencia 300Hz y ciclo de trabajo 0,1

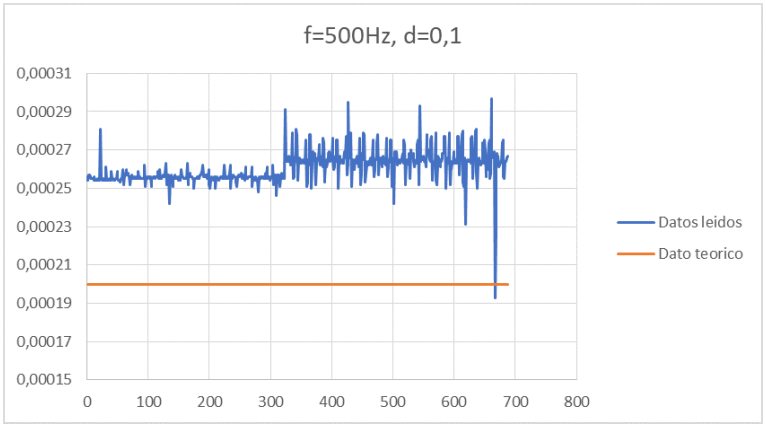


Fig. 8. Lectura de PWM con frecuencia 500Hz y ciclo de trabajo 0,1